

IDENTIFICATION

PRODUCT ID: ZZ-ESKAB-14.0

PRODL. TITLE: ESKAB140 VAX 11/780 MICRO DIAGNOSTIC MONITOR

DECO/DEPO: 14.0

DATE: NOV 1982

MAINTAINED BY: VAX DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1977,1982

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 10754

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MEY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFCRMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

VAX 11/780 MICRO DIAGNOSTIC USER'S GUIDE

REV 0 9 MAR 1977 D. MONROE INITIAL RELEASE
REV 1 14 APR 1977 D. MONROE
REV 2 10 JUN 1977 D. MONROE
ADDED THE FOLLOWING FEATURES:
EXAMINE ID BUS REGISTERS
LOOP ON SPECIAL TEST (LT) AND SECTION (LS) FLAGS
REV 3 29 JUL 1977 D. MONROE
ADDED THE FOLLOWING FEATURES:
EXAMINE INTERNAL REGISTERS
EXAMINE VBUS CHANNELS
CLOCK SPEED CONTROL
REV 4 14 OCT 1977 D. MONROE
REV 5 15 DEC 1977 D. MONROE
ADDED DEPOSIT FUNCTION
ADDED CONTINUE SWITCH TO DIAG COMMAND
ADDED "SET/CLR FPSYNC" COMMAND
REV 6 22 FEB 1978 D. MONROE
ADDED CAPABILITY TO SPECIFY A SPAN OF TESTS
OR SECTIONS TO THE 'DIAGNOSE' COMMAND
REV 7 30 JUN 1978 D. MONROE
ADDED CAPABILITY OF COMMENTS AND STRINGS OF
SPACES IN COMMAND LINES
CHANGED INTERNAL MESSAGE FORMATS TO RADIX 50
REV 8 24 OCT 1978 D. MONROE
REPARTITIONED THE TWO FLOPPIES
MODIFIED THE MEMORY TESTS
ADDED INFORMATION TO SECTION 4.3 OF THIS DOCUMENT
ADDED SCRATCHPAD DUAL ADDRESSING TEST TO HARDCORE
ADDED A PSEUDO INSTRUCTION "SPAGEN"
FIXED BUG REPORTED IN PROBLEM REPORT #FFF67
REV 9 7 DEC 1978 D. MONROE
ADDED TYPEOUT OF SYSTEM ID REGISTER TO SECTION 20.
ADDED TEST OF M8213 TO SECTION 4E.
ADDED 'EXAMINE/DEPOSIT' FUNCTION FOR SBI ADDRESSES. (SEE
PARAGRAPH 2.7 AND 2.8 OF THIS DOCUMENT)
ADDED 'REPEAT' FUNCTION TO ALL COMMANDS (SEE PARAGRAPH
2.10 OF THIS DOCUMENT).

ZZ-ESKAB-14.0 Documentation
REV 11 31 MAY 1979

D 1

Fiche 1 Frame D1

Sequence 3

D. MONROE

FIXED EXAMINES OF ID BUS REGISTER 18 (SBI SILO) SO
THEY NOW WORK. NOTICE: SILO MUST BE LOCKED BY THE MICRO
TEST BEFORE EXAMINE.

DELETED TESTING OF THE SLOW CONSTANT ROM IN THE HARDCORE
TESTS.

ADDED FLOPPY 3 FOR THE MULTIPORT MEMORY TO THE MICRO
DIAGNOSTIC PACKAGE. SEE SECTION 4.6 OF THIS DOCUMENT
FOR MORE INFORMATION.

REPARTITIONED FLOPPIES 1 AND 2 AGAIN. FLOPPY 1 NOW
CONTAINS SECTIONS 1 THRU 3D. FLOPPY 2 CONTAINS
SECTIONS 3E THRU 5D, AND FLOPPY 3 (MA780) CONTAINS
SECTIONS 3E THRU 63.

ADDED A SECTION TO THIS DOCUMENT (4.7) THAT EXPLAINS
HOW TO INTERPRET THE CACHE DATA FOR THE INSTRUCTION
IFER AND FPA TESTS.

REV 11.1 13 AUG 1979 D. MONROE, V. DILL, T. VEZZA

ESKAJ

ADDED A TEST TO SECTION 3E TO TEST THE LENGTH OF THE
TIMEOUT COUNTER. ADDED TWO NEW SECTIONS (SECTION 41
AND SECTION 51) TO INCREASE THE TEST COVERAGE ON THE
SBL AND SBH MODULES. THE FLOATING POINT SECTIONS ARE
NOW NUMBERED 53 THRU 5F.

ESKAN

ADDED 2 NEW SECTIONS TO (MA780 FLOPPY) TO TEST
CPU TO CPU CONFIGURATIONS.

REV 11.2 14 DEC 1979 D. MONROE, V. DILL, T. VEZZA

ESKAC - ALLOWED FOR 4K OF WCS.

ESKAD - ADDED TEST COVERAGE FOR 4K OF WCS

ESKAJ - GENERAL BUG FIXES

ESKAN - GENERAL BUG FIXES

REV 11.3 15 FEB 1980 D. MONROE

ESKAC - ADDED A PSEUDO INSTRUCTION FOR FPA VBUS TEST

ESKAD - ADDED A TEST FOR THE FPA VBUS

ESKAH - FIXED VARIOUS BUGS (SEE RELEASE NOTES)

ESKAJ - FIXED VARIOUS BUGS (SEE RELEASE NOTES)

ESKAN - FIXED TEST IN SECTION 45 THAT FAILED BECAUSE
OF A HARDWARE DESIGN PROBLEM THAT ISN'T GOING
TO BE FIXED.

REV 12.0 16 JUN 1980 D. MONROE

ESKAB - ADDED NEW MACRO'S 'GETMDM' AND 'LDCNSL' TO
SUPPORT EUROPEAN REMOTE DIAGNOSIS

ESKAN - FIXED BUG IN SECTION 65 TO SUPPORT 3 AND 4
PROCESSOR CONFIGURATIONS.

ZZ-ESKAB-14.0 Documentation
REV 13.0 4 MAY 1981

E 1

Fiche 1 Frame E1

Sequence 4

D. MONROE
ESKAB - UPDATED FOR FLOPPY RE-PARTITION
ESKAC - ADDED NEW MACRO 'CHKKEY' TO ALLOW CONSOLE TO
ESKAE CHECK KEYSWITCH DURING MICRO EXECUTION.
ESKAH - REPARTITIONED (NOW CONTAINS SECTIONS 20 - 3C)
ADDED TWO NEW TESTS FOR 'TWINKLE' ECO
ESKAJ - REPARTITIONED (NOW CONTAINS SECTIONS 3D - 5F)
ADDED NEW TEST PATTERNS FOR 'TWINKLE' ECO
ESKAL - REPARTITIONED (NOW CONTAINS SECTIONS 20 - 3C)
ESKAM - REPARTITIONED (NOW CONTAINS SECTIONS 3D - 5F)
ESKAN - REPARTITIONED (NOW CONTAINS SECTIONS 3D - 65)
ESKAP - REPARTITIONED (NOW CONTAINS SECTIONS 3D - 65)

REV 13.1 16 DEC 1981 B. LANDRY/B. POLAND
REV 13.1 16 DEC 1981 B. LANDRY/B. POLAND
ESKAB - MODIFIED TO RETURN TO MIC PROMPT AT END OF
MICRO 2. WCS FILE DELETED FROM THIS FLOPPY.

REV 14.0 SEPTEMBER 1981 DON MONROE
ESKAB - MODIFIED TO SUPPORT THE MS780-E FLOPPY (MICRO
3). MOVED SINGLE INSTRUCTION ROUTINE TO
HARDCORE MONITOR AND REPLACED THE TRAP 46
CALL WITH AN ENABLE CONTROL C FUNCTION.

CONTENTS

- 1.0 INTRODUCTION
- 2.0 COMMANDS
 - 2.1 DIAGNOSE
 - 2.2 CONTINUE
 - 2.3 SET AND CLR
 - 2.4 SHOW
 - 2.5 LOOP
 - 2.6 RETURN
 - 2.7 EXAMINE
 - 2.8 DEPOSIT
 - 2.9 CONTROL C
 - 2.10 REPEAT
- 3.0 QUALIFIERS
- 4.0 RUNNING THE MICRO DIAGNOSTICS
 - 4.1 CONSOLE COMMANDS
 - 4.2 MICRO DIAGNOSTIC COMMAND
 - 4.3 ERROR REPORTS
 - 4.4 SECTION PARTITIONING
 - 4.5 SBI DEVICE TESTS
 - 4.6 MULTI-PORT MEMORY (MA780) TESTS
 - 4.7 CACHE DATA
- 5.0 SYNTAX ERROR MESSAGES
 - 5.1 ?USE DIAG COMMAND
 - 5.2 ?INVALID COMMAND
 - 5.3 ?INVALID KEYWORD
 - 5.4 ?NUMBER MUST BE HEX
- 6.0 SYSTEM ERROR MESSAGES
 - 6.1 ?OPEN FILE
 - 6.2 ?READ SECTOR
 - 6.3 ?KEYBOARD ERROR
 - 6.4 ?UNEXPECTED TRAP TO 4...PC=
- 7.0 GO CHAIN MONITOR ERROR MESSAGES
 - 7.1 ?TIMEOUT IN TEST UPC=
 - 7.2 ?EXECUTION OUT OF SEQUENCE
 - 7.3 ?CLOCK STOPPED UNEXPECTEDLY UPC=
 - 7.4 ?ILLEGAL MONITOR CALL UPC=

ZZ-ESKAB-14.0 Documentation
8.0 GO CHAIN INFORMATION MESSAGES
8.1 CPU TR=
8.2 MS780 4K CHIP AT TR XX
8.3 MS780 16K CHIP AT TR XX
8.4 MAX ADR+1=
8.5 DEVXX AT TR XX
8.6 SYS ID REG=
8.7 # SINGLE BIT ERRORS:
8.8 ERROR LOG LIMIT EXCEEDED:
8.9 TEST ABORTFD - NO DW780 AVAILABLE
8.10 TEST ABORTED - NOT ENOUGH MEMORY
8.11 M8213 ROMS OK
8.12 ?NO M8213 ROMS
8.13 NO FPA
8.14 STARTING FPA TESTS
8.15 DEPOSIT CTRLR ADRS IN RCO & TYPE LO
8.16 KE780 FPLA PRESENT/NOT PRESENT

THIS DOCUMENT GIVES A BRIEF DESCRIPTION OF THE COMMANDS AVAILABLE IN THE MICRO DIAGNOSTIC MONITOR. ALTHOUGH ALL COMMANDS, KEYWORDS, QUALIFIERS, AND FLAGS ARE SPELLED OUT, THEY CAN ALL BE ABBREVIATED TO THE FIRST TWO CHARACTERS EXCEPT FOR THE FLAGS "HALTI" AND "HALTD" WHICH ARE ABBREVIATED "HI" AND "HD" RESPECTIVELY.

ALSO DESCRIBED HERE IS SOME GENERAL INFORMATION ABOUT THE 11/780 MICRO DIAGNOSTIC PACKAGE.

2.0

COMMANDS

2.1 DIAGNOSE COMMAND

SYNTAX: DIAG

QUALIFIERS: SEE SECTION 3.0

SEMANTICS: INITIALIZES THE PROGRAM CONTROL FLAGS AND STARTS EXECUTION OF THE MICRO DIAGNOSTICS AT TEST NUMBER 1.

2.2 CONTINUE COMMAND

SYNTAX: CONT

SEMANTICS: CONTINUES EXECUTION OF THE MICRO DIAGNOSTICS WITHOUT CHANGING THE PROGRAM CONTROL FLAGS.

THIS COMMAND CAN ONLY BE USED TO CONTINUE EXECUTION OF THE TESTS. IT CANNOT BE USED TO INITIATE EXECUTION. ALSO, IF AN EXAMINE/DEPOSIT SBI COMMAND WAS USED, THIS COMMAND CANNOT BE USED TO CONTINUE EXECUTION.

AN ERROR MESSAGE IS TYPED IF THE COMMAND IS USED INCORRECTLY.

2.3 SET AND CLR COMMAND

SYNTAX: SET(OR CLR) FLAG <LIST OF FLAGS SEPARATED BY COMMAS>

SET(OR CLR) SOMM:<ADDRESS>

SET(OR CLR) FPSYNC:<ADDRESS>

SET STEP STATE

SET STEP BUS

SET STEP INSTRUCTION

SET CLOCK (FAST, SLOW, EXTERNAL OR NORMAL)

SEMANTICS:

'SET FLAG' AND 'CLR FLAG' SEMANTICS:

SET [CLR] FLAG HALTD

SETS (OR CLEARS) THE HALT ON ERROR DETECTION FLAG.

SET [CLR] FLAG HALTI

SETS OR CLEARS THE HALT ON ERROR ISOLATION FLAG.

SET [CLR] FLAG LOOP

SETS OR CLEARS THE LOOP ON ERROR FLAG

SET [CLR] FLAG NER

SETS OR CLEARS THE NO ERROR REPORT FLAG

SET [CLR] FLAG BELL
SETS OR CLEARS THE BELL ON ERROR FLAG

SET [CLR] FLAG ERABT
SETS OR CLEARS THE ERROR ABORT FLAG

CLR FLAG LS
CLEAR THE LOOP ON SPECIAL SECTION FLAG

CLR FLAG LT
CLEAR THE LOOP ON SPECIAL TEST FLAG

SET [CLR] FLAG ALL
SETS OR CLEARS ALL OF THE ABOVE FLAGS.

NOTE: THE LS AND LT FLAGS CANNOT BE "SET".

'SET [CLR] SOMM' SEMANTICS:

SET [CLR] SOMM
SETS OR CLEARS THE STOP ON MICRO MATCH BIT

SET [CLR] SOMM:<ADDRESS>
LOADS <ADDRESS> INTO THE MICRO BREAK
REGISTER AND SETS OR CLEARS THE STC? ON MICRO
MATCH BIT.

USING THE CLR SOMM:<ADDRESS> COMMAND ALLOWS A
SYNC PULSE TO BE GENERATED EVERYTIME MICROCODE
IS EXECUTED AT <ADDRESS>. MICROCODE EXECUTION
WILL NOT STOP. THE SYNC PULSE IS OUTPUT ON A
TEST CONNECTOR LOCATED ON SIDE 1 (ABOUT 3
INCHES FROM THE TOP) OF THE M8235 MODULE.

USING THE SET SOMM:<ADDRESS> COMMAND CAUSES
MICROCODE EXECUTION TO STOP AT THE <ADDRESS>
SPECIFIED.

'SET [CLR] FPSYNC' SEMANTICS:

SET [CLR] FPSYNC:<ADDRESS>
LOADS <ADDRESS> INTO THE FPA MICRO SYNC REGISTER

'SET STEP STATE' AND 'SET STEP BUS' SEMANTICS:

SET STEP STATE
SETS THE CLOCK TO SINGLE TIME STATE

SET STEP BUS
SETS THE CLOCK TO SINGLE BUS CYCLE

BOTH OF THE ABOVE TWO COMMANDS ENTER STEP MODE.
STEP MODE TYPES THE CURRENT STATE OF THE CLOCK OR
VALUE OF THE UPC REGISTER
AND WAITS FOR TERMINAL INPUT. IF A SPACE IS TYPED,
THE CLOCK IS TICKED AND THE CURRENT VALUE OF THE UPC
IS TYPED AGAIN. IF ANY OTHER CHARACTER IS TYPED,
STEP MODE IS EXITED.

'SET STEP INSTRUCTION' SEMANTICS:

SET STEP INSTRUCTION

SETS THE HARDCORE SINGLE INSTRUCTION FLAG
AND RETURNS TO THE MICRO MONITOR.

WHEN THE HARDCORE TESTS ARE INVOKED, THE CURRENT VALUE
OF THE TEST PC (TPC) IS TYPED AND TERMINAL INPUT IS WAITED FOR.
IF A SPACE IS TYPED, THE CURRENT PSEUDO INSTRUCTION IS EXECUTED
AND THE CURRENT VALUE OF THE TPC IS TYPED AGAIN. IF ANY
OTHER CHARACTER IS TYPED, STEP MODE IS EXITED.

'SET CLOCK SEMANTICS:

SET CLOCK FAST

SETS THE CPU CLOCK SPEED TO THE FAST MARGIN

SET CLOCK SLOW

SETS THE CPU CLOCK SPEED TO THE SLOW MARGIN

SET CLOCK NORMAL

SETS THE CPU CLOCK SPEED TO NORMAL

SET CLOCK EXTERNAL

SETS THE CPU CLOCK FOR AN EXTERNAL OSCILLATOR

2.4 SHOW COMMAND

SYNTAX: SHOW

SEMANTICS: DISPLAYS THE HALTD, HALTI, LOOP, NER,
BELL, ERABT, LS, AND LT FLAGS.

2.5 LOOP COMMAND

SYNTAX: LOOP

SEMANTICS: CLEARS THE HALTI AND HALTD FLAGS, SETS
THE LOOP AND NER FLAGS, AND EXECUTES A CONTINUE COMMAND.

2.6 RETURN COMMAND

SYNTAX: RETURN

SEMANTICS: RETURNS TO THE CONSOLE PROGRAM.

2.7 EXAMINE COMMAND

SYNTAX: EXAMINE ID:<ADDRESS>
EXAMINE VBUS:<CHANNEL>
EXAMINE RA:<ADDRESS>
EXAMINE RC:<ADDRESS>
EXAMINE LA
EXAMINE LC
EXAMINE DR
EXAMINE QR
EXAMINE SC
EXAMINE FE
EXAMINE VA
EXAMINE PC
EXAMINE SBI:<ADDRESS>

SEMANTICS:

EXAMINE ID:<ADDRESS>
DISPLAYS THE ADDRESS AND THE CONTENTS OF THE ID BUS REGISTER SPECIFIED BY <ADDRESS>.

EXAMINE VBUS:<CHANNEL>
DISPLAYS THE CHANNEL NUMBER AND THE CONTENTS OF THE VBUS CHANNEL SPECIFIED BY <CHANNEL>. BIT 0 IS AT THE RIGHT HAND SIDE OF THE DISPLAY.

EXAMINE RA:<ADDRESS>
DISPLAYS THE CONTENTS OF THE RA SCRATCH PAD SPECIFIED BY <ADDRESS>.

EXAMINE RC:<ADDRESS>
DISPLAYS THE CONTENTS OF THE RC SCRATCH PAD SPECIFIED BY <ADDRESS>.

EXAMINE LA
DISPLAYS THE CONTENTS OF THE LA LATCH.

EXAMINE LC
DISPLAYS THE CONTENTS OF THE LC LATCH.

EXAMINE DR
DISPLAYS THE CONTENTS OF THE D REGISTER

EXAMINE QR
DISPLAYS THE CONTENTS OF THE Q REGISTER

EXAMINE SC
DISPLAYS THE CONTENTS OF THE SC REGISTER

EXAMINE FE
DISPLAYS THE CONTENTS OF THE FE REGISTER

EXAMINE VA
DISPLAYS THE CONTENTS OF THE VA REGISTER

EXAMINE PC

DISPLAYS THE CONTENTS OF THE PROGRAM COUNTER REGISTER

EXAMINE SBI:<ADDRESS>

DISPLAYS THE ADDRESS AND THE DATA IN THE ADDRESS.

NOTE: ANY OF THESE EXAMINES CAUSES THE CURRENT MICRO INSTRUCTION TO BE EXECUTED BEFORE THE EXAMINE IS PERFORMED IF IT IS THE FIRST EXAMINE SINCE ENTERING THE MICRO DIAGNOSTIC MONITOR. ALL SUCCESSIVE EXAMINES DO NOT EXECUTE ANY MORE MICRO INSTRUCTIONS. ID BUS REGISTERS T5 THRU T0D ARE DESTROYED DURING THE ABOVE EXAMINES EXCEPT FOR V BUS EXAMINES. ALL OFF THE ABOVE EXAMINES, EXCEPT VBUS, ADVANCE THE CLOCK TO CPTO BEFORE EXECUTING THE EXAMINE.

THE EXAMINE SBI COMMAND PERFORMS A PHYSICAL LONG WORD READ ON THE SBI. IF A BYTE ADDRESS IS SPECIFIED, THE LOWER ORDER TWO BITS ARE CLEARED BEFORE THE EXAMINE. THE CACHE AND TRANSLATION BUFFER ARE SHUT OFF BEFORE THE EXAMINE. IF THE EXAMINE TIMES OUT, AN ERROR MESSAGE IS TYPED AND THE TIMEOUT BIT IN THE SBI ERROR REGISTER IS CLEARED.

2.8 DEPOSIT COMMAND

SYNTAX: DEPOSIT ID:<ADDRESS> <DATA>
DEPOSIT RA:<ADDRESS> <DATA>
DEPOSIT RC:<ADDRESS> <DATA>
DEPOSIT LA:<DATA>
DEPOSIT LC:<DATA>
DEPOSIT DR:<DATA>
DEPOSIT QR:<DATA>
DEPOSIT SC:<DATA>
DEPOSIT FE:<DATA>
DEPOSIT VA:<DATA>
DEPOSIT PA:<DATA>
DEPOSIT SBI:<ADDRESS> <DATA>

SEMANTICS:

THE DEPOSIT COMMAND IS THE SAME AS THE EXAMINE COMMAND WITH THE ADDITION OF THE NUMBER TO DEPOSIT.

A DEPOSIT TO THE SBI IS NOT AUTOMATICALLY LONG WORD ALIGNED. IF A BYTE ADDRESS IS SPECIFIED, ONLY THE UPPER BYTES OF THE LONG WORD ARE WRITTEN.

2.9 CONTROL C

IF TYPED ON A COMMAND LINE, IT WILL ABORT THE COMMAND.

IF TYPED WHILE THE MICRO DIAGNOSTICS ARE RUNNING, THE CURRENTLY EXECUTING TEST WILL COMPLETE AND "COMMAND MODE" WILL BE ENTERED.

IF TYPED WHILE LOOPING ON AN ERROR, THE LOOP WILL BE SUSPENDED AND COMMAND MODE ENTERED.

2.10 REPEAT

SYNTAX: R <COMMAND STRING>

SEMANTICS:

IF AN "R" IS TYPED BEFORE THE NORMAL COMMAND STRING, THE COMMAND WILL BE REPEATED UNTIL A CONTROL C IS TYPED. FOR EXAMINE COMMANDS, THE TYPEOUT CAN BE KILLED BY TYPING CONTROL O.

ALL COMMANDS CAN BE REPEATED EXCEPT FOR THE FOLLOWING:

SET
CLEAR
EXAMINE VBUS
DIAGNOSE
CONTINUE
LOOP
RETURN

QUALIFIERS (SWITCHES) MAY BE USED WITH THE 'DIAGNOSE' COMMAND TO SPECIFY LOOPING ON A PARTICULAR TEST OR OVERLAY, OR TO OVERRIDE THE DEFAULT PASS COUNT.

VALID QUALIFIERS:

- /TEST:<NUMBER> -- DISPATCH TO THE TEST NUMBER SPECIFIED (DON'T EXECUTE ANY PRIOR TESTS) AND LOOP ON THE TEST INDEFINITELY.
- /SECT:<NUMBER> -- DISPATCH TO THE SECTION NUMBER SPECIFIED (DON'T EXECUTE ANY PRIOR SECTIONS) AND LOOP ON THE SECTION INDEFINATELY.
- /PASS:<NUMBER> -- EXECUTE THE MICRO DIAGNOSTICS THE SPECIFIED NUMBER OF PASSES BEFORE RETURNING TO THE CONSOLE. IF THE NUMBER IS "-1", EXECUTE THE MICRO DIAGNOSTICS INDEFINATELY.
- /CONTINUE -- THIS SWITCH IS USED WITH THE /TEST OR /SECT SWITCH TO AUTOMATICALLY CONTINUE AFTER THE SPECIFIED TEST OR SECTION HAS BEEN REACHED.
- /TEST:<N> <M> -- DISPATCH TO TEST <N>, EXECUTE TESTS <N> THROUGH <M> (INCLUSIVE), AND RETURN TO COMMAND MODE.
- /SECT:<N> <M> -- DISPATCH TO SECTION <N>, EXECUTE SECTIONS <N> THROUGH <M> (INCLUSIVE), AND RETURN TO COMMAND MODE.

NOTE: IN THE ABOVE TWO VARIATIONS OF THE "/TEST" AND "/SECTION" QUALIFIERS, THE VALUE OF <N> MUST BE LESS THAN OR EQUAL TO <M>. IF <M> IS LESS THAN <N>, TESTING WILL START AT <N> AND CONTINUE TO THE END.

NOTE: /TEST AND /SECT CANNOT BE SPECIFIED SIMULTANEOUSLY.

EXAMPLES:

- DIAG/TEST:2F
DISPATCH TO TEST NUMBER 2F AND EXECUTE IT INDEFINITELY.
- DIAG/SECT:B
DISPATCH TO SECTION NUMBER B AND EXECUTE IT INDEFINITELY.
- DIAG/PASS:-1
EXECUTE ALL OF THE MICRO DIAGNOSTICS INDEFINITELY.
- DIAG/TEST:2F/CONT
DISPATCH TO TEST 2F AND START EXECUTION OF THE REMAINING TESTS.

4.1 CONSOLE COMMANDS

TO INVOKE THE MICRO DIAGNOSTICS FROM THE CONSOLE PROGRAM,
THE "TEST" COMMAND IS USED. IF THE MICRO DIAGNOSTIC MONITOR
IS REQUIRED BEFORE BEGINNING EXECUTION OF THE MICRO DIAGNOSTICS
THE "TEST" COMMAND WITH THE "COMMAND" QUALIFIER IS USED.
THAT IS, "TEST/COM" WOULD BE THE COMMAND.

4.2 MICRO DIAGNOSTIC COMMAND

ONCE IN THE MICRO DIAGNOSTIC MONITOR, THE "DIAGNOSE" OR "CONTINUE"
COMMAND IS USED TO START TESTING. SEE SECTION 2.1 AND 2.2 FOR
FURTHER DETAILS OF THESE COMMANDS.

4.3 ERROR REPORTS

THE ERROR REPORT FOR BOTH THE HARDCORE AND GO CHAIN TESTS
IS IDENTICAL EXCEPT FOR THE "<ERROR PC>". THIS NUMBER IS TYPED
IN "OCTAL" FOR THE HARDCORE TESTS AND "HEXIDECLIMAL" FOR THE
GO CHAIN TESTS. THE HARDCORE TESTS ARE SECTIONS 1 THRU 1F AND
TESTS 1 THRU 3F.

FOLLOWING IS THE FORMAT OF THE ERROR REPORT:

```
<SECTION NUMBER>, <SECTION NUMBER>, ...  
?ERROR:<ERROR PC> TEST:<TEST NUMBER> SUBTEST:<SUB TEST NUMBER>  
DATA: <HEXIDECIMAL NUMBER>  
      <HEXIDECIMAL NUMBER>  
      :  
      :  
TRACE: <TRACE ID>, <TRACE ID>, ...  
FAILING MODULES: <MODULE NAME>, <MODULE NAME>, ...
```

THE <SECTION NUMBER> INDICATES WHICH SECTION THE ERROR OCCURRED
IN. I.E., THE LAST NUMBER TYPED BEFORE THE ERROR, IS THE SECTION
THAT THE FAILING TEST IS IN.

THE <ERROR PC> IS EITHER THE "TEST PC" (FOR A HARDCORE
FAILURE) OR THE "MICRO PC" FOR A GO CHAIN FAILURE.

THE <TEST NUMBER> AND <SUB TEST NUMBER> ARE SELF EXPLANATORY.

THE <HEXIDECIMAL NUMBER> IS DEFINED BY THE PROGRAMMER. A LISTING
OF THE FAILING TEST (OR ESKAL.SEQ/ESKAM.SEQ) IS
REQUIRED TO INTERPRET THIS DATA.

THE <TRACE ID> IS USED TO IDENTIFY WHICH "FAIL CHAIN" ENTRIES
WERE USED TO ISOLATE THE FAILURE. THIS TRACE CAN BE USED TO
DETERMINE WHICH V BUS SIGNALS WERE EXAMINED AND THE STATE THEY
WERE FOUND TO BE IN. THIS INFORMATION IS CONTAINED IN THE
DOCUMENTS ESKAL.SEQ AND ESKAM.SEQ.

THE <MODULE NAME> IS THE 'M82...' NUMBER OF THE FAILING MODULE OR MODULES. THE MODULES (IF THERE ARE MORE THAN ONE LISTED) ARE LISTED IN THE MOST PROBABLE ORDER FROM HIGHEST TO LOWEST PROBABILITY. IN SOME HARDCORE TESTS, A BUS NAME IS ALSO TYPED (SUCH AS 'BUSID', 'BUSVB', OR 'BUSCS') WHICH INDICATES THAT THE FAILURE COULD BE ON ANY MODULE CONNECTED TO THIS BUS.

SECTIONS 4E AND 4F MAY ALSO TYPE AN ADAPTER NAME (SUCH AS 'UBA' OR 'MBA') SINCE THESE SECTIONS USE THE ADAPTERS TO TEST LOGIC IN THE CPU. SECTION 50 MAY ALSO TYPE AN ADAPTER NAME SINCE IT TESTS THE SBI ERROR LOGIC IN THE ADAPTERS.

4.4 SECTION PARTITIONING

THE "HARDCORE TESTS" TEST THE CONSOLE ADAPTER, THE MICRO SEQUENCER, THE WCS AND PCS, AND PART OF THE DATA PATH. THE GO CHAIN TESTS ARE PARTITIONED INTO 9 MAJOR CATEGORIES AS FOLLOWS:

- 1) DATA PATH TESTS
- 2) CACHE MEMORY TESTS
- 3) TRANSLATION BUFFER TESTS
- 4) INSTRUCTION BUFFER TESTS
- 5) CONDITION CODES, INTERRUPTS, AND EXCEPTIONS TESTS
- 6) SBI INTERFACE TESTS
- 7) MEMORY TESTS
- 8) SBI DEVICE TESTS
- 9) FLOATING POINT ACCELERATOR TESTS
- 10) MULTI-PORT MEMORY TESTS

CATAGORIES 1 THRU 5 ARE PACKAGED ON FLOPPY NUMBER 1 (SECTIONS 1 THRU 3D), CATAGORIES 6 THRU 9 ON FLOPPY NUMBER 2 (SECTIONS 3E THRU 5F), AND CATAGORY 10 (SECTIONS 3E THRU 63) ON FLOPPY 3.

FLOPPY 3 IS ONLY REQUIRED FOR SYSTEMS WITH AN MA780. FLOPPY 2 IS ONLY REQUIRED FOR SYSTEMS WITH AN MS780 OR FP780.

4.5 SBI DEVICE TESTS

CATAGORY 8, OF THE GO CHAIN, USES ANY AVAILABLE DEVICES (UBA'S OR MBA'S) THAT ARE FOUND ON THE SBI TO TEST THEIR FAULT DETECTION LOGIC. CATAGORY 8 ALSO USES A UBA (IF THERE IS ONE ON THE SYSTEM) TO TEST THE CACHE INVALIDATION LOGIC. THE INVALIDATE TEST ALSO REQUIRES 192K BYTES OF MEMORY.

FLOPPY 3 (FILES ESKAN AND ESKAP) CONTAIN MICRO DIAGNOSTIC TESTS FOR THE MA780 MULTI-PORT MEMORY. THIS FLOPPY ALSO CONTAINS CPU/SBI INTERFACE AND SBI NEXUS TESTS THAT ARE DUPLICATED FROM FLOPPY 2 AND MODIFIED TO USE THE MA780 INSTEAD OF THE MS780. THESE TESTS ARE INCLUDED SO THAT FLOPPY 3 CAN SUPPORT SYSTEMS WITHOUT AN MS780. NOTE HOWEVER, THAT THE FPA TESTS ARE ONLY INCLUDED ON FLOPPY 2.

THE STARTING SECTION AND TEST NUMBERS ON FLOPPY 3 ARE THE SAME AS FLOPPY 2 (SECTION 3E, TEST 178). NOTE THAT THIS IS A CHANGE FROM REVISION 9.X.

INCLUDED ON THIS FLOPPY ARE SOME SPECIAL TESTS:

- A. DESIGNED TO RUN ON A CPU WITH MORE THAN ONE PORT CONNECTED TO A PARTICULAR MA780 (USED IN VOLUME MANUFACTURING).
- B. DESIGNED TO RUN SIMULTANEOUSLY ON MORE THAN ONE CPU CONNECTED TO A PARTICULAR MA780 (TO BE USED BY FA&T AND FIELD SERVICE).
- C. DESIGNED TO TEST POWER UP/DOWN LOGIC IN THE MA780.

SECTIONS 3E(X) THRU 62(X) CAN BE RUN USING NORMAL MICRO DIAGNOSTIC OPERATING PROCEDURES, E.G. EITHER OF THE COMMANDS:

```
>>>T <CR>
OR
MIC>DI <CR>
```

WILL RESULT IN THE EXECUTION OF SECTIONS 3E THRU 62. NOTE THAT THESE SECTIONS (3E THRU 62), ARE NOT CPU TO CPU INTERACTION TESTS.

4.6.1 MA780 POWER FAIL TESTS

SECTION 63 OF FLOPPY 3 IS A POWER FAIL TEST OF THE MA780. THIS SECTION IS SKIPPED BY DEFAULT. TO PERFORM THIS SECTION REQUIRES MANUAL INTERVENTION. THE FOLLOWING SEQUENCE IS REQUIRED:

1. POWER DOWN, THEN POWER UP THE MA780.
2. TYPE 'DI SE:63 CO' TO THE MICRO DIAGNOSTIC MONITOR.
3. WHEN THE MESSAGE 'PWR FAIL PORT' IS TYPED, POWER FAIL THE MA780 AGAIN.
4. THE DIAGNOSTIC WILL TYPE SOME INFORMATION MESSAGES SUCH AS POWER UP STARTING ADDRESS AND NUMBER OF SINGLE BIT ERRORS IN THE ARRAY CARDS AND INVALIDATE MAP.
5. IF ALL TESTS WERE SUCCESSFUL, THE MESSAGE 'PWR FAIL OK' IS TYPED.
6. IF MORE THAN ONE PORT IS ON THE SYSTEM, RETURN TO STEP 3.

IT IS VIRTUALLY IMPOSSIBLE TO "LOOP ON ERROR" IN THIS SECTION BECAUSE OF THE REQUIREMENT FOR MANUAL INTERVENTION. REFER TO THE LISTING OF THE MICRO CODE TO DETERMINE WHICH LOGIC IS BEING TESTED.

SECTIONS 64(X) AND 65(X) CONTAIN TESTS DESIGNED TO RUN IN MORE THAN ONE CPU SIMULTANEOUSLY. ANY OR ALL OF THESE TESTS CAN BE STARTED UP ON 2, 3, OR 4 CPU'S EACH CONNECTED TO THE SAME 1, 2, 3, OR 4 MA780'S. TO START THEM RUNNING TYPE:

MIC>DI SE:64 CO

ON THE CONSOLE'S OF EACH CPU WHICH WILL BE USED TO TEST THE MA780'S IN THE SYSTEM. IF A PARTICULAR PORT/CPU IS NOT WANTED BY THE OPERATOR TO BE INVOLVED, THAT PORT MUST BE SWITCHED OFF LINE. IT IS POSSIBLE TO LOOP ON A SPECIFIC TEST OR SECTION WITH THE FOLLOWING COMMANDS:

MIC>DI TE:<TEST NUMBER>

OR

MIC>DI SE:<SECTION NUMBER>

NOTE THAT THE STARTUP COMMAND USED MUST BE THE SAME ON ALL CPU'S.

THE OPERATOR HAS ABOUT 10 MINUTES TO GO FROM CONSOLE TERMINAL TO CONSOLE TERMINAL FOR THIS STARTUP PROCEDURE. IF MORE THAN 10 MINUTES IS TAKEN, THE FIRST CPU STARTED WILL REPORT AN ERROR. DURING THIS STARTUP TIME, THE FIRST CPU'S STARTED WILL WAIT FOR ALL THE OTHER ON LINE CPU'S TO COME UP BEFORE RUNNING ANY TESTS.

THE OPERATOR SHOULD HAVE A SEPARATE FLOPPY DISK CONTAINING THIS DIAGNOSTIC FOR EACH CPU INVOLVED. THESE SECTIONS WILL NOT EXECUTE CORRECTLY ON SYSTEMS CONTAINING A CPU WITH MORE THAN ONE PORT INTO A PARTICULAR MA780. IF THE SYSTEM IS CONFIGURED THUS, USE THE OFF LINE SWITCHES TO RECONFIGURE.

4.6.3 APT/APT-RD INTERFACE

THE MICRO DIAGNOSTIC MONITOR WILL AUTOMATICALLY LOOK FOR THE FILES "ESKAJ" AND "ESKAN" (IN THAT ORDER) AFTER RUNNING "ESKAH". IF "ESKAJ" IS FOUND, "ESKAN" WILL NOT BE EXECUTED AND VISA VERSA. THEREFORE, FOR AN APT SCRIPT TO TEST AN MA780 SYSTEM, THE FILE "ESKAJ" MUST NOT BE ON THE VIRTUAL FLOPPY.

MOST OF THE INSTRUCTION BUFFER TESTS (SECTIONS 33 THRU 3C) AND SOME OF THE FPA TESTS REQUIRE DATA TO BE FETCHED FROM THE CACHE. THE DATA IS AUTOMATICALLY LOADED INTO THE CACHE WHEN THE SECTION OF MICRO CODE IS LOADED INTO WCS. THE DATA THAT IS LOADED APPEARS IN THE LISTING IN THE FOLLOWING FORMAT:

```
:% <HEX ADDRESS>
:$ <16 BIT HEX NUMBER>,<16 BIT HEX NUMBER>,...  
:$ <16 BIT HEX NUMBER>,<16 BIT HEX NUMBER>,...
```

THE NUMBER THAT FOLLOWS THE ";%" IS THE STARTING ADDRESS OF THE DATA. THE NUMBERS THAT FOLLOW A ";\$" ARE GROUPED INTO 16 BIT FIELDS SEPARATED BY COMMAS. THE DATA IS SEQUENTIALLY WRITTEN INTO THE CACHE STARTING AT THE STARTING ADDRESS. CONSIDER THE FOLLOWING EXAMPLE:

THE LISTING LOOKS AS FOLLOWS:

```
:% 200
:$ 1234,5678,9ABC,DEF0
```

THIS DATA WOULD GET LOADED INTO THE CACHE AS FOLLOWS:

LONG WORD ADDRESS	DATA BITS
	31-----0
200	56781234
204	DEF09ABC

IN OTHERWORDS, THE FIRST 16 BIT NUMBER IS THE LEAST SIGNIFICANT 16 BITS OF THE FIRST LONGWORD. THE SECOND IS THE MOST SIGNIFICANT 16 BITS. THE THIRD, THE LEAST SIGNIFICANT 16 BITS OF THE SECOND LONG WORD AND THE FOURTH, THE MOST SIGNIFICANT 16 BITS OF THE SECOND LONG WORD.

THE DATA IS ALWAYS LOADED INTO GROUP 0 FIRST AND IF THERE ARE MORE THAN 1024 LONGWORDS OF DATA, GROUP 1 IS SELECTED (STARTING AT AT ADDRESS 1000(X)) FOR THE REST OF THE DATA.

5.1 ?USE DIAG COMMAND -- A CONTINUE COMMAND TRIED TO BE
EXECUTED BEFORE A DIAGNOSE COMMAND.
THIS WILL ONLY OCCUR IF 'TEST/COM' WAS
USED TO INVOKE THE MICRO DIAGNOSTICS FROM THE CONSOLE PROGRAM.

5.2 ?INVALID COMMAND -- THE COMMAND WAS NOT RECOGNIZED.

5.3 ?INVALID KEYWORD -- AN ARGUMENT OF ANY COMMAND WAS UNRECOGNIZED.

5.4 ?NUMBER MUST BE HEX -- A NON HEXIDECLIMAL NUMBER WAS RECOGNIZED.

6.1 ?OPEN FILE:<NUMBER> -- ERROR WHEN TRYING TO OPEN A FLOPPY FILE.
<NUMBER>=2 -- FILE NOT FOUND
<NUMBER>=3 -- FLOPPY QUEUE FULL

6.2 ?READ SECTOR:<NUMBER> -- ERROR WHEN TRYING TO READ A SECTOR
FROM THE FLOPPY.
<NUMBER>=4 -- SECTOR # OUT OF RANGE
<NUMBER>=3 -- FLOPPY QUEUE FULL
<NUMBER>=1 -- ERROR, CRC OR PARITY

6.3 ?KEYBOARD ERROR:<NUMBER> -- ERROR WHEN TRYING TO READ THE TERMINAL
<NUMBER>=5 -- TERMINAL DRIVER BUSY
<NUMBER>=7 -- ERROR

6.4 ?UNEXPECTED TRAP TO 4...PC= -- THE LSI-11 TRAPED TO 4 AT THE
SPECIFIED PC.

7.1 ?TIMEOUT IN TEST ___ UPC= -- THE MONITOR DID NOT RECEIVE A CALL FROM THE MICRO CODE IN THE LAST 4 SECONDS. INDICATES THE MICRO CODE IS HUNG.

7.2 ?EXECUTION OUT OF SEQUENCE
UPC=____ SHOULD BE=____ -- THE MICRO CODE HAS NOT EXECUTED THE TESTS WITHIN THE OVERLAY IN SEQUENTIAL ORDER. TYPING 'CO' WILL RESTART EXECUTION IN THE TEST THAT CAUSED THE OUT OF SEQUENCE.

7.3 ?CLOCK STOPPED UNEXPECTEDLY -- THE CLOCK STOPPED AND THE "SOMM" BIT WAS NOT SET.

7.4 ?ILLEGAL MONITOR CALL:<NUMBER> -- THE MICRC CODE MADE A CALL TO THE MONITOR WITH A BAD ARGUMENT WHICH WAS <NUMBER>.

THE FOLLOWING MESSAGES ARE TYPED WHILE THE MICRO DIAGNOSTICS ARE RUNNING. SOME OF THEM ARE SYSTEM CONFIGURATION INFORMATION WHILE OTHERS ARE CONFIGURATION ERRORS. ANY CONFIGURATION THAT IS A FATAL ERROR ALSO TYPES AN ERROR MESSAGE.

- 8.1 CPU TR=
THIS MESSAGE IDENTIFIES WHAT THE TR LEVEL OF THE CPU IS. IN MOST CASES IT SHOULD ALWAYS BE 10(H).
- 8.2 MS780 4K CHIP AT TR XX
THIS MESSAGE IDENTIFIES MEMORY CONTROLLERS, THE ARRAY SIZE, AND THE TR LEVEL OF THE CONTROLLER.
- 8.3 MS780 16K CHIP AT TR XX
THIS IS THE SAME AS THE PREVIOUS MESSAGE EXCEPT FOR THE ARRAY SIZE.
- 8.3.5 MA780 AT TR XX
IDENTIFIES MULTI PORT MEMORY CONTROLLERS AND THEIR TR LEVEL.
- 8.4 MAX ADR+1=
THIS MESSAGE IDENTIFIES THE MAXIMUM ADDRESS OF A MEMORY CONTROLLER BASED ON THE CONTENTS OF CONFIGURATION REGISTER B. THIS MESSAGE WILL ALWAYS IMMEDIATELY FOLLOW MESSAGE 8.2, 8.3, OR 8.3.5.
- 8.4.5 PORT #:
THIS MESSAGE WILL ALWAYS BE PRECEDED BY MESSAGE 8.3.5. IT IDENTIFIES THE PORT NUMBER OF AN MA780 AT THE SPECIFIED TR LEVEL.
- 8.5 DEVXX AT TR XX
THIS MESSAGE IDENTIFIES OTHER DEVICES ON THE SBI. CURRENTLY THE PROGRAM RECOGNIZES DW780'S, RH780'S, DR780'S, CI780'S, AND MS780-E'S. IT DISPLAYS THE TR OF THE DEVICE.
- 8.6 SYS ID REG=
THIS MESSAGE TYPES THE CONTENTS OF THE SYSTEM ID REGISTER.
- 8.7 # OF SINGLE BIT ERRORS:
THIS MESSAGE IS USED TO IDENTIFY THE NUMBER OF SINGLE BIT ERRORS IN A MEMORY ARRAY. IT IS ALWAYS PRECEDED BY A MESSAGE THAT IDENTIFIES THE CONTROLLER AND A MESSAGE THAT IDENTIFIES THE ARRAY NUMBER ON THAT CONTROLLER.
- 8.8 ERROR LOG LIMIT EXCEEDED:
THIS MESSAGE IS USED TO INDICATE THAT A MEMORY ARRAY CONTAINS MORE THAN 1000(D) SINGLE BIT ERRORS.
- 8.9 TEST ABORTED - NO DW780 AVAILABLE
THIS MESSAGE IS TYPED IF THE SYSTEM DOES NOT HAVE A DW780. SOME CPU TESTING CANNOT BE PERFORMED WITHOUT A DW780.
- 8.10 TEST ABORTED - NOT ENOUGH MEMORY
THIS MESSAGE IS USED TO INDICATE THAT THE TEST OF PART OF THE SBI INVALIDATE LOGIC IS NOT BEING TESTED BECAUSE THERE IS NOT ENOUGH MEMORY ON THE SYSTEM. THE TEST REQUIRES AT LEAST 192K BYTES.

THIS MESSAGE MEANS THAT THE BOOT STRAP ROMS ARE CONFIGURED CORRECTLY AND THAT THE CHECK SUM IS OK. THIS MESSAGE IS ALWAYS PRECEDED BY MESSAGE 8.2 OR 8.3 TO INDICATE WHICH CONTROLLER THE ROMS ARE LOCATED ON.

IF ROMS ARE FOUND ON BOTH CONTROLLERS (OF A TWO CONTROLLER SYSTEM), AN ERROR MESSAGE IS TYPED SINCE IT IS AN ILLEGAL CONFIGURATION.

8.12 ?NO M8213 ROMS

THIS MEANS THAT THERE ARE NO BOOT STRAP ROMS ON THE SYSTEM. MESSAGE 8.15 AND AN ERROR MESSAGE IS TYPED IMMEDIATELY AFTER THIS MESSAGE.

8.13 NO FPA

THIS MESSAGE MEANS THAT THE ACCELERATOR STATUS REGISTER DOES NOT HAVE BIT 0 SET.

8.14 STARTING FPA TESTS

THIS MESSAGE INDICATES THE START OF FPA TESTING.

8.15 DEPOSIT CTRLR ADRS IN RCO & TYPE LO

THIS MESSAGE IS TO INDICATE THE ABILITY TO LOOP THE READ OF AN SBI ADDRESS SPECIFIED BY THE OPERATOR. THIS MESSAGE IS TYPED IF A MEMORY CONTROLLER DOES NOT RESPOND OR IF M8213 ROMS DO NOT RESPOND.

8.16 KE780 FPLA PRESENT [NOT PRESENT]

ONE OF TWO MESSAGES IS TYPED DEPENDING ON WHETHER THE FPLA FOR THE KE780 FLOATING POINT OPTION IS INSTALLED.

VAX 11/780 MICRO DIAGNOSTIC

HARDCORE INSTRUCTION DICTIONARY

AUTHOR: DONALD W MONROE

REVISION A	APRIL 7, 1977
REVISION B	APRIL 14, 1977
REVISION C	JULY 29, 1977
REVISION D	DECEMBER 15, 1977
REVISION E	OCTOBER 24, 1978

THIS DOCUMENT DESCRIBES THE PSEUDO INSTRUCTIONS USED IN THE HARDCORE TEST STREAM. EACH INSTRUCTION IS SHOWN WITH ITS OPCODE NAME FOLLOWED BY THE ARGUMENTS FOR THE OPCODE. ARGUMENT NAMES ARE ENCLOSED IN ANGLE BRACKETS (<>). OPTIONAL ARGUMENTS, THOSE THAT ARE NOT ABSOLUTELY REQUIRED, ARE ENCLOSED IN SQUARE BRACKETS ([]).

LEGAL INDEX NAMES ARE: I, J, OR K.

IF THE <MODE> ARGUMENT (SEE THE CMPXXX INSTRUCTIONS) IS NOT SPECIFIED, IT DEFAULTS TO "EQUAL".

IF THE <REGISTER> ARGUMENT TO A CMPXXX INSTRUCTION IS SPECIFIED AS "'IDREGLO'" OR "'IDREGHI'", THE REGISTER USED IN THE COMPARE IS THE ID BUS REGISTER THAT WAS READ IN THE MOST PREVIOUS "READID" INSTRUCTION.

BLKMIC [<SRC ADDRESS>], [<SRC INDEX>], <WCS ADDRESS>, <WORD COUNT>. [<WCS ADDRESS INDEX>]

MOVE <WORD COUNT> NUMBER OF 96 BIT MICRO WORDS FROM <SRC ADDRESS>, INDEXED BY <SRC INDEX>, TO WCS STARTING AT <WCS ADDRESS>, INDEXED BY <WCS ADDRESS INDEX>. IF <SRC INDEX> IS SPECIFIED, THE <SRC ADDRESS> IS INDEXED BY 6 PDP-11 WORDS (96 BITS).

IF THE <WCS ADDRESS> STARTS WITH AN ALPHA CHARACTER, THE <WCS ADDRESS> IS USED AS A POINTER TO A TABLE IN LSI-11 MEMORY OTHERWISE, IT IS USED AS THE PHYSICAL WCS ADDRESS.

FOR EXAMPLE: IF THE CURRENT VALUE OF THE INDEX IS 2, 14(8) (<SRC INDEX> * 6) WOULD BE ADDED TO THE <SRC ADDRESS> TO FIND THE FIRST 96 BIT MICRC WORD TO LOAD INTO WCS.

CHKPNT [<PASS ADDRESS>], [<FAIL ADDRESS>]

IF THE ERROR FLAG (SEE THE CMPXXX INSTRUCTIONS) IS ZERO, GO TO THE <PASS ADDRESS>. IF THE ERROR FLAG IS NON ZERO, GO TO THE <FAIL ADDRESS>. IF THE ADDRESSE(S) IS NOT SPECIFIED, GO TO THE NEXT, INLINE, INSTRUCTION.

THE ADDRESS OF THE NEXT INSTRUCTION IS TYPED. THESE ADDRESSES APPEAR ON THE TYPED LINE NAMED "TRACE:".

CLOCK <TIMES>

TICK THE SYSTEM CLOCK <TIMES> NUMBER OF SINGLE TIME STATES. IF <TIMES> IS AN INTEGER NUMBER OF 4, SINGLE BUS CYCLES ARE EXECUTED FOR EACH 4 <TIMES>.

CMPCA [<MODE>], <REGISTER>, <DST ADDRESS>, [<DST ADDRESS INDEX>]

COMPARE THE CONSOLE ADAPTER REGISTER SPECIFIED BY <REGISTER> WITH THE CONTENTS OF THE LOCATION SPECIFIED BY <DST ADDRESS>, INDEXED BY <DST ADDRESS INDEX>. IF THE <MODE> IS FALSE, SET THE ERROR FLAG.

CMPCAD [<MODE>], <REGISTER>, <DST ADDRESS>, [<DST ADDRESS INDEX>]

COMPARE THE CONTENTS OF THE CONSOLE ADAPTER REGISTERS SPECIFIED BY <REGISTER> AND <REGISTER>+2 WITH THE CONTENTS OF THE LOCATIONS SPECIFIED BY <DST ADDRESS> AND <DST ADDRESS>+2 (INDEXED BY <DST ADDRESS INDEX>).

IF THE <MODE> IS FALSE, SET THE ERROR FLAG

CMPCAM [<MODE>],<REGISTER>,<MASK ADDRESS>,[<MASK ADDRESS INDEX>],<DST ADDRESS>,[<DST ADDRESS INDEX>]

TAKE THE CONTENTS OF THE CONSOLE ADAPTER REGISTER SPECIFIED BY <REGISTER>, MASK IT WITH THE CONTENTS OF THE <MASK ADDRESS> (INDEXED BY <MASK ADDRESS INDEX>), AND COMPARE IT WITH THE CONTENTS OF THE <DST ADDRESS> (INDEXED BY <DST ADDRESS INDEX>).

IF THE <MODE> IS FALSE, SET THE ERROR FLAG.

THE MASK IS PERFORMED BY TAKING THE CONTENTS OF <MASK ADDRESS> (INDEXED BY <MASK ADDRESS INDEX>), COMPLIMENTING IT, AND BIT CLEARING THE CONTENTS OF <REGISTER> WITH IT.

CMPCMD [<MODE>],<REGISTER>,<MASK ADDRESS>,[<MASK ADDRESS INDEX>],<DST ADDRESS>,[<DST ADDRESS INDEX>]

TAKE THE CONTENTS OF THE CONSOLE ADAPTER REGISTERS SPECIFIED BY <REGISTER> AND <REGISTER>+2, MASK IT WITH THE CONTENTS OF THE <MASK ADDRESS> AND <MASK ADDRESS>+2 (INDEXED BY <MASK ADDRESS INDEX>), AND COMPARE IT WITH THE CONTENTS OF THE <DST ADDRESS> AND <DST ADDRESS>+2 (INDEXED BY <DST ADDRESS INDEX>).

IF THE <MODE> IS FALSE, SET THE ERROR FLAG.

THE MASK IS PERFORMED BY TAKING THE CONTENTS OF <MASK ADDRESS> AND <MASK ADDRESS>+2 (INDEXED BY <MASK ADDRESS INDEX>), COMPLIMENTING IT, AND BIT CLEARING THE CONTENTS OF <REGISTER> AND <REGISTER>+2 WITH IT.

CMPPCSV <DST ADDRESS>,[<DST ADDRESS INDEX>]

COMPARE THE CONTENTS OF THE PC SAVE REGISTER WITH THE CONTENTS OF THE LOCATION SPECIFIED BY <DST ADDRESS> (INDEXED BY <DST ADDRESS INDEX>). IF THEY ARE NOT EQUAL, SET THE ERROR FLAG.

ENDLOOP <INDEX NAME>

ADD THE INCREMENT VALUE OF THIS <INDEX NAME> (SEE THE "LOOP" INSTRUCTION) TO THE CURRENT VALUE OF THE INDEX SPECIFIED BY <INDEX NAME>. COMPARE THE CURRENT VALUE WITH THE LAST VALUE (SPECIFIED IN THE "LOOP" INSTRUCTION). IF THE CURRENT VALUE IS LESS THAN OR EQUAL TO THE LAST VALUE, GO TO THE INSTRUCTION FOLLOWING THE MOST RECENT "LOOP" INSTRUCTION (WITH THE SAME <INDEX NAME>). OTHERWISE, GO TO THE NEXT SEQUENTIAL I INSTRUCTION.

SAVE THE ADDRESS OF THE NEXT INSTRUCTION.

IF AN ERROR IS DETECTED AND "LOOP ON ERROR" IS SELECTED (SEE THE MICRO DIAGNOSTIC USERS MANUAL), EXECUTION IS RESTARTED AT THIS SAVED ADDRESS AFTER THE "IFERROR" INSTRUCTION IS EXECUTED.

FETCH <WCS ADDRESS>,[<WCS ADDRESS INDEX>],[<INHIBIT ROM NOP>]

IF <WCS ADDRESS> IS A NUMERIC STRING, THEN EXECUTE A MAINTENANCE RETURN TO THE LOCATION SPECIFIED BY <WCS ADDRESS> (INDEXED BY <WCS ADDRESS INDEX>).

IF <WCS ADDRESS> IS AN ALPHA-NUMERIC STRING THEN, EXECUTE A MAINTENANCE RETURN TO THE LOCATION SPECIFIED BY THE CONTENTS OF <WCS ADDRESS> (INDEXED BY <WCS ADDRESS INDEX>).

IF <INHIBIT ROM NOP> IS DEFINED THEN CLEAR "ROM NOP" DURING THE FETCH OF THE SPECIFIED ADDRESS OTHERWISE, DO NOT CHANGE THE STATE OF "ROM NOP".

FILLWCS <SRC ADDRESS>

FILL THE CONTENTS OF THE 5TH AND 6TH K OF WCS ADDRESS SPACE WITH THE CONTENTS OF THE MICRO WORD SPECIFIED BY <SRC ADDRESS>.

FLTONE <DST ADDRESS>,<INDEX NAME>

GENERATE A 32 BIT WORD OF ALL ZERO'S. INSERT A ONE IN THE BIT POSITION SPECIFIED BY THE CURRENT VALUE MINUS 1 OF <INDEX NAME>, AND PUT THIS WORD IN THE LOCATION SPECIFIED BY <DST ADDRESS> AND <DST ADDRESS>+2.

FLTZRO <DST ADDRESS>,<INDEX NAME>

GENERATE A 32 BIT WORD OF ALL ONE'S. INSERT A ZERO IN THE BIT POSITION SPECIFIED BY THE CURRENT VALUE MINUS 1 OF <INDEX NAME>, AND PUT THIS WORD IN THE LOCATION SPECIFIED BY <DST ADDRESS> AND <DST ADDRESS>+2.

IFERROR [<MESSAGE NUMBER>],[<FAIL ADDRESS>]

IF THE ERROR FLAG IS NON ZERO, TYPE THE PC OF THIS INSTRUCTION,
THE TEST NUMBER, AND THE SUBTEST NUMBER, AND THE GOOD AND BAD
DATA THEN, GO TO <FAIL ADDRESS> IF THE 'HALTD' FLAG
(SEE THE MICRO DIAGNOSTIC USERS MANUAL) IS NOT SET.

IF THE ERROR FLAG IS ZERO OR THE <FAIL ADDRESS> IS NOT
SPECIFIED, GO TO THE NEXT INSTRUCTION.

INITIALIZE

SET AND CLEAR THE CPU INITIALIZE BIT IN THE MACHINE CONTROL
REGISTER, CLEAR THE SINGLE TIME STATE BIT, SET THE SINGLE BUS
CYCLE BIT, SET THE ROM NOP BIT, AND SET THE PROCEED BIT IN THE MACHINE CONTROL REGISTER.

KMXGEN <SRC ADDRESS>,<INDEX NAME>

GENERATE THE KMUX ADDRESS SPECIFIED BY THE CURRENT VALUE MINUS 1
OF <INDEX NAME> AND PUT IT IN THE KMUX FIELD OF THE MICRO INSTRUCTION
SPECIFIED BY <SRC ADDRESS>.

LDIDREG <REGISTER>,<SRC ADDRESS>,[<SRC ADDRESS INDEX>]

LOAD THE ID BUS REGISTER SPECIFIED BY <REGISTER> WITH THE CONTENTS
OF THE LOCATIONS SPECIFIED BY <SRC ADDRESS> AND <SRC ADDRESS>+2
(INDEXED BY <SRC ADDRESS INDEX>).

IF <REGISTER> IS THE 'MICRO STACK', 'MICRO BREAK', OR 'WCS ADDRESS',
THE CONTENTS OF <SRC ADDRESS> IS TAKEN TO BE 16 BITS. OTHERWISE,
IT IS TAKEN TO BE 32 BITS.

LOADCA <REGISTER>,<SRC ADDRESS>,[<SRC ADDRESS INDEX>]

LOAD THE CONSOLE ADAPTER REGISTER SPECIFIED BY <REGISTER> WITH THE
CONTENTS OF THE LOCATION SPECIFIED BY <SRC ADDRESS> (INDEXED BY
<SRC ADDRESS INDEX>).

THIS INSTRUCTIONS ONLY LOADS 16 BITS OF DATA.

ZZ-ESKAB-14.0 Documentation
LOOP <INDEX NAME>,<START>,<END>,[<SIZE DEPENDENT>]

E 3

Fiche 1 Frame E3

Sequence 30

INITIALIZE THE LOOP PARAMETER SPECIFIED BY <INDEX NAME> TO THE VALUE SPECIFIED BY <START>. SAVE THE VALUE SPECIFIED BY <END> FOR THE 'ENDLOOP' INSTRUCTION. CALCULATE AND SAVE THE INCREMENT VALUE FOR THE 'ENDLOOP' INSTRUCTION WITH THE FOLLOWING ALGORITHM:

IF <START> IS LESS THAN OR EQUAL TO <END>, SET THE INCREMENT VALUE TO +1 OTHERWISE, SET IT TO -1.

IF <END> IS AN <INDEX NAME>, SAVE THE CURRENT VALUE OF THAT INDEX NAME AS THE <END> VALUE OF THIS INDEX NAME.

IF <SIZE DEPENDENT> IS SPECIFIED, DIVIDE THE LARGER OF <START> AND <END> BY TWO IF THERE IS ONLY ONE WCS MODULE ON THE SYSTEM OTHERWISE, LEAVE THEM UNCHANGED.

MASK <DST ADDRESS>,<MASK ADDRESS>

TAKE THE CONTENTS OF LOCATION <MASK ADDRESS>, COMPLIMENT IT, AND BIT CLEAR THE CONTENTS OF LOCATION <DST ADDRESS> WITH IT.

MOVE <SRC ADDRESS>,[<SRC INDEX>],<DST ADDRESS>,[<DATA TYPE>]

MOVE THE CONTENTS OF LOCATION <SRC ADDRESS>, INDEXED BY <SRC INDEX>, TO THE LOCATION SPECIFIED BY <DST ADDRESS>. IF <DATA TYPE> IS SPECIFIED, THE INDEX IS A 32 BIT INDEX, OTHERWISE IT IS A 16 BIT INDEX.

NEWTST <TEST NAME>,[<TEST DESCRIPTION>],[<LOGIC DESCRIPTION>],[<ERROR DESCRIPTION>],[<SYNC POINT DESCRIPTION>]

CREATES A TEST HEADER DOCUMENT FROM THE SPECIFIED ARGUMENTS.

CLEAR THE ERROR FLAG. SAVES THE PC OF THE NEXT INSTRUCTION FOR LOOPING ON TEST (SEE THE MICRO DIAGNOSTIC USERS MANUAL).

READID <REGISTER>

READS THE ID BUS REGISTER SPECIFIED BY <REGISTER> AND PUTS THE CONTENTS OF IT IN LOCATIONS "IDREGLO" AND "IDREGHI".

RESET

EXECUTE AN LSI-11 RESET INSTRUCTION

TYPE THE MODULE NUMBERS OF THE MODULES SPECIFIED BY <MODULE NAME STRING>.

IF THE HALTI FLAG IS SET, RETURN TO THE MICRO DIAGNOSTIC MONITOR (SEE THE MICRO DIAGNOSTIC USERS MANUAL).

TSTVB <SRC TABLE ADDRESS>,[<SRC TABLE ADDRESS INDEX>]

LOAD AND READ THE V BUS. COMPARE THE CONTENTS OF THE DATA AT <SRC TABLE ADDRESS> (INDEXED BY <SRC TABLE ADDRESS INDEX>) WITH THE V BUS DATA JUST READ.

THE <SRC TABLE> HAS THE FOLLOWING FORMAT:

1\$: .WORD <NUMBER OF BITS TO CHECK>
VBUSG <CHANNEL NUMBER>,<BIT NUMBER>,<EXPECTED BIT VALUE>

2\$: .WORD <NUMBER OF BITS TO CHECK>
VBUSG <CHANNEL NUMBER>,<BIT NUMBER>,<EXPECTED BIT VALUE>

.

.

.

FOLLOWING IS AN EXAMPLE OF THE <SRC TABLE ADDRESS INDEX>:

TSTVB 1\$,I

IF THE CURRENT VALUE OF THE <SRC TABLE ADDRESS INDEX> IS 2,
AND THE <SRC TABLE> LOOKS LIKE THE ABOVE TABLE, THE PHYSICAL
<SRC TABLE ADDRESS> WOULD BE 2\$.

SETPSW <DATA>

LOAD THE LSI-11 PROCESSOR STATUS WORD WITH THE VALUE SPECIFIED BY <DATA>.

SETVEC <VECTOR ADDRESS>

SET THE PDP-11 ADDRESS SPECIFIED BY <VECTOR ADDRESS> TO THE EXPECTED TRAP ROUTINE.

SPAGEN <SRC ADDRESS>,<INDEX NAME>

GENERATE THE SCRATCH PAD ADDRESS SPECIFIED BY THE CURRENT VALUE MINUS 1 OF <INDEX NAME> AND PUT IT IN THE SPA FIELD OF THE MICRO INSTRUCTION SPECIFIED BY <SRC ADDRESS>.

GO TO THE <DST ADDRESS>. IF <DST ADDRESS> IS NOT SPECIFIED, GO TO
THE NEXT TEST. IF <DST ADDRESS> STARTS WITH THE ALPHA CHARACTER "S",
GO TO THE NEXT SUB TEST.

SKIPERROR

SKIP THE REST OF THE TEST IF THE ERROR FLAG IS SET.

SUBTEST

INCREMENT THE SUB TEST COUNTER.

TYPSIZE

USE THE CONTENTS OF LOCATION "BADDATA" TO DETERMINE THE WCS MODULE
CONFIGURATION AND TYPE A MESSAGE AND THE NUMBER OF WCS MODULES THAT
WILL BE TESTED.

IF THE WCS MODULE COUNT IS ZERO OR IF BITS 3 THRU 0 ARE NON ZERO,
OR IF THE 5TH K OF WCS IS NOT PRESENT,
THE TEST STREAM IS ABORTED AND THE MICRO DIAGNOSTIC MONITOR IS
ENTERED UNLESS THE NER FLAG IS SET.

VAX 11/780 VISIBILITY BUS DIRECTORY

ZZ-ESKAB-14.0 Documentation

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
00	0000	00000	USCF	M8235	CPTO	USCF UPCSV 00 H
00	0001	00001	USCF	M8235	CPTO	USCF UPCSV 01 H
00	0002	00002	USCF	M8235	CPTO	USCF UPCSV 03 H
00	0003	00003	USCF	M8235	CPTO	USCF UPCSV 03 H
00	0004	00004	USCF	M8235	CPTO	USCF UPCSV 04 H
00	0005	00005	USCF	M8235	CPTO	USCF UPCSV 05 H
00	0006	00006	USCF	M8235	CPTO	USCF UPCSV 06 H
00	0007	00007	USCF	M8235	CPTO	USCF UPCSV 07 H
00	0008	00010	USCF	M8235	CPTO	USCF UPCSV 08 H
00	0009	00011	USCF	M8235	CPTO	USCF UPCSV 09 H
00	000A	00012	USCF	M8235	CPTO	USCF UPCSV 10 H
00	000B	00013	USCF	M8235	CPTO	USCF UPCSV 11 H
00	000C	00014	USCF	M8235	CPTO	USCF UPCSV 12 H
00	000D	00015	USCB	M8235	CPTX	USCB STALL H
00	000E	00016	USCB	M8235	CPTX	USCB UTRAP H
00	000F	00017	USCJ	M8235	CPTX	USCJ ECO DISPATCH 06 H
00	0010	00020	USCE	M8235	CPT1	USCE ID BUS XCVR EN L
00	0011	00021	USCE	M8235	CPT3	USCE CS WR (31:00) H
00	0012	00022	USCE	M8235	CPT3	USCE CS WR (63:32) H
00	0013	00023	USCE	M8235	CPT3	USCE CS WR (95:64) H
00	0014	00024	USCE	M8235	CPTX	USCE WCS WR CYCLE H
00	0015	00025	USCE	M8235	CPT1	USCE WCS MEM AVAIL L
00	0016	00026	USCL	M8235	CPTX	FCTX ACC OVERRIDE L
00	0017	00027	USCM	M8235	CPTX	JSCM IBUF EN (07:00) L
00	0018	00030	USCN	M8235	CPTX	A
00	0019	00031	USCN	M8235	CPTX	ICLK ALU Z (1) H
00	001A	00032	USCN	M8235	CPTX	DCPA LA00 H
00	001B	00033	USCN	M8235	CPTX	D
00	001C	00034	USCN	M8235	CPTX	E
00	001D	00035	USCN	M8235	CPTX	F
00	001E	00036	USCN	M8235	CPTX	ACCA UB0 H
00	001F	00037	USCN	M8235	CPTX	J
00	0020	00040	USCN	M8235	CPTX	K
00	0021	00041	USCN	M8235	CPTX	CEH3 PSL C BIT H
00	0022	00042	USCN	M8235	CPTX	ICLK ALU C (1) H
00	0023	00043	USCN	M8235	CPTX	N
00	0024	00044	USCN	M8235	CPTX	P
00	0025	00045	USCN	M8235	CPTX	ACCA UB1 H
00	0026	00046	USCN	M8235	CPTX	S
00	0027	00047	USCN	M8235	CPTX	T
00	0028	00050	USCN	M8235	CPTX	DCPA LA01 H
00	0029	00051	USCN	M8235	CPTX	V
00	002A	00052	USCN	M8235	CPTX	X
00	002B	00053	USCN	M8235	CPTX	Y
00	002C	00054	USCN	M8235	CPTX	ACCA UB2 H
00	002D	00055	USCN	M8235	CPTX	CEHE UTRAP VECT 0 H
00	002E	00056	USCN	M8235	CPTX	TBMD LAST REF CODE 1 H
00	002F	00057	USLN	M8235	CPTX	DAPD SS(1) H
00	0030	00060	USCN	M8235	CPTX	DD
00	0031	00061	USCN	M8235	CPTX	CEHE UTRAP VECT 1 H
00	0032	00062	USCN	M8235	CPTX	TBMD LAST REF CODE 0 H
00	0033	00063	USCN	M8235	CPTX	DDPS SC N.E. 0 H
00	0034	00064	USCN	M8235	CPTX	JJ

00	0035	00065	USCN	M8235	CPTX	CEHE UTRAP VECT 2 H
00	0036	00066	USCN	M8235	CPTX	CEHF NESTED ERR (1) H
00	0037	00067	USCN	M8235	CPTX	ICLK EALU Z (1) H
00	0038	00070	USCN	M8235	CPTX	NN
00	0039	00071	USCN	M8235	CPTX	CEHE UTRAP VECT 3 H
00	003A	00072	USCN	M8235	CPTX	CEHH FPD BIT L
00	003B	00073	USCN	M8235	CPTX	ICLK EALU N (1) H
00	003C	00074	USCN	M8235	CPTX	TT
00	003D	00075	USCN	M8235	CPTX	USCN BEN EN (1B:14) H
00	003E	00076	USCN	M8235	CPTX	USCN BEN EN (1F:1C) H
00	003F	00077	USCN	M8235	CPTX	USCN BEN EN (13:10) H
00	0040	00100	USCN	M8235	CPTX	USCN BEN EN (07:00) H
00	0041	00101	USCN	M8235	CPTX	USCN BEN EN (0F:08) H
00	0042	00102	USCP	M8235	CPTX	USCP BRBIT0(1F:1C) H
00	0043	00103	USCP	M8235	CPTX	ICLE BRBIT0(1B:14) H
00	0044	00104	USCP	M8235	CPTX	ICLE BRBIT0(0F:08) H
00	0045	00105	USCP	M8235	CPTX	USCP BRBIT1(1F:1C) H
00	0046	00106	USCP	M8235	CPTX	ICLE BRBIT1(1B:14) H
00	0047	00107	USCP	M8235	CPTX	ICLE BRBIT1(0F:08) H
00	0048	00110	USCP	M8235	CPTX	USCP BRBIT2(1F:1C) H
00	0049	00111	USCP	M8235	CPTX	ICLE BRBIT2(1B:14) H
00	004A	00112	USCP	M8235	CPTX	ICLE BRBIT2(0F:08) H
00	004B	00113	USCP	M8235	CPTX	USCP BRBIT3(1F:1C) H
00	004C	00114	USCP	M8235	CPTX	ICLE BRBIT3(1B:14) H
00	004D	00115	USCP	M8235	CPTX	USCP BRBIT4(1F:1C) H
00	004E	00116	USCH	M8235	CPT3	USCH SYNC PULSE H
00	004F	00117	USCJ	M8235	CPT0	CIBN D MAINT RTN H
00	0050	00120	USCJ	M8235	CPTX	USCJ INIT (1) H
00	0051	00121	USCJ	M8235	CPTX	USCJ STALL (1) H
00	0052	00122	USCJ	M8235	CPTX	USCJ UTRAP (1) H
00	0053	00123	USCJ	M8235	CPTX	USCJ UECO (1) H
00	0054	00124	USCJ	M8235	CPTX	USCJ MAINT RET (1) H
00	0055	00125	USCJ	M8235	CPTX	USCJ PRIOR 0 L
00	0056	00126	USCJ	M8235	CPTX	USCJ PRIOR 1 L
00	0057	00127	USCJ	M8235	CPTX	USCJ PRIOR 2 L
00	0058	00130	USCM	M8235	CPT2	USCM BUF UPC 00 H
00	0059	00131	USCM	M8235	CPT2	USCM BUF UPC 01 H
00	005A	00132	USCM	M8235	CPT2	USCM BUF UPC 02 H
00	005B	00133	USCM	M8235	CPT2	USCM BUF UPC 03 H
00	005C	00134	USCM	M8235	CPT2	USCM BUF UPC 04 H
00	005D	00135	USCM	M8235	CPT2	USCM BUF UPC 05 H
00	005E	00136	USCM	M8235	CPT2	USCM BUF UPC 06 H
00	005F	00137	USCM	M8235	CPT2	USCM BUF UPC 07 H
00	0060	00140	USCM	M8235	CPT2	USCM BUF UPC 08 H
00	0061	00141	USCM	M8235	CPT2	USCM BUF UPC 09 H
00	0062	00142	USCM	M8235	CPT2	USCM BUF UPC 10 H
00	0063	00143	USCM	M8235	CPT2	USCM BUF UPC 11 H
00	0064	00144	USCM	M8235	CPT2	USCM BUF UPC 12 H
00	0065	00145	USCX	M8235	CPTX	RESERVED
00	0066	00146	USCX	M8235	CPTX	RESERVED
00	0067	00147	USCX	M8235	CPTX	RESERVED

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
01	0000	00000	CEHE	M8230	CPTX	PCSC PAR ERR (95:64) H
01	0001	00001	CEHE	M8230	CPTX	PCSC PAR ERR (63:32) H
01	0002	00002	CEHE	M8230	CPTX	PCSC PAR ERR (31:0C) H
01	0003	00003	CEHE	M8230	CPTX	SELP PAR ERR TRAP L
01	0004	00004	CEHE	M8230	CPTX	TBMW PROT UTRAP L
01	0005	00005	CEHF	M8230	CPTX	CEHF IRD STATE H
01	0006	00006	CEHF	M8230	CPTX	CEHF READ RLOG H
01	0007	00007	CEHF	M8230	CPTX	CEHF LOAD STATE H
01	0008	00010	CEHF	M8230	CPTX	CEHF CLR UWORD (?) H
01	0009	00011	CEHP	M8230	CPTX	ICLS EN ID XCEIV L
01	000A	00012	CEHA	M8230	CPTX	DEPM BMX31 L
01	0008	00013	CEHA	M8230	CPTX	DDPD BMX15 L
01	000C	00014	CEHA	M8230	CPTX	DCPD PMX07 L
01	000D	00015	CEHA	M8230	CPTX	DEPD AMX31 L
01	000E	00016	CEHA	M8230	CPTX	DDPB AMX15 L
01	000F	00017	CEHA	M8230	CPTX	DCPD AMX07 L
01	0010	00020	CEHA	M8230	CPTX	DEPK ALU BUFF31 L
01	0011	00021	CEHA	M8230	CPTX	DCPL ALU CARRY31 L
01	0012	00022	CEHA	M8230	CPTX	DCPL ALU CARRY15 L
01	0013	00023	CEHA	M8230	CPTX	DCPL ALU CARRY07 L
01	0014	00024	CEHA	M8230	CPTX	CEHA ALU BUFF17 L
01	0015	00025	CEHA	M8230	CPTX	CEHA ALU BUFF16 L
01	0016	00026	CEHA	M8230	CPTX	CEHA ALU BUFF15 L
01	0017	00027	CEHA	M8230	CPTX	CEHA ALU BUFF07 L
01	0018	00030	CEHA	M8230	CPTX	DEPN ALU(30:18)=0 L
01	0019	00031	CEHA	M8230	CPTX	DDPF ALU(15:08)=0 L
01	001A	00032	CEHA	M8230	CPTX	DCPF ALU(07:00)=0 L
01	001B	00033	CEHA	M8230	CPTX	BUS ALU BYTE2,3 A=B H
01	001C	00034	CEHA	M8230	CPTX	BUS ALU BYTE1 A=B H
01	001D	00035	CEHA	M8230	CPTX	BUS ALU BYTE0 A=B H
01	001E	00036	CEHB	M8230	CPTX	DCPA AMX00 L
01	001F	00037	CEHB	M8230	CPTX	DAPB AUALU=A PLUS B L
01	0020	00040	CEHB	M8230	CPTX	DAPB AUALU=A MINUS B L
01	0021	00041	CEHC	M8230	CPTX	DDPN EALU09 H
01	0022	00042	CEHC	M8230	CPTX	DDPN EALU08 H
01	0023	00043	CEHC	M8230	CPTX	ACCX NDATA H
01	0024	00044	CEHC	M8230	CPTX	ACCX ZDATA H
01	0025	00045	CEHC	M8230	CPTX	ACCX VDATA H
01	0026	00046	CEHC	M8230	CPTX	ACCX CDATA H
01	0027	00047	CEHD	M8230	CPTX	CEHD SECOND REF H
01	0028	00050	CEHD	M8230	CPTX	SBLT STALL L
01	0029	00051	CEHD	M8230	CPTX	IRCJ DQ CONT H
01	002A	00052	CEHD	M8230	CPTX	IRCJ FLOAT H
01	002B	00053	CEHD	M8230	CPTX	IRCJ WORD CONT H
01	002C	00054	CEHD	M8230	CPTX	IRCJ BYTE CONT H
01	002D	00055	CEHD	M8230	CPTX	TBMW SAVE CONTEXT H
01	002E	00056	CEHE	M8230	CPTX	DDPS FLOAT NZERO H
01	002F	00057	CEHE	M8230	CPTX	USC3 CLR UTRAP L
01	0030	00060	CEHR	M8230	CPTX	DCPH VA02(1) H
01	0031	00061	CEHR	M8230	CPTX	DCPJ VA01(1) H

ZZ-ESKAB-14.0 Documentation

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
------	--------------	----------------	-----	--------	------	-------------

01	0032	00062	CEHR	M8230	CPTX	DCPJ VA00(1) H
01	0033	00063	CEHE	M8230	CPTX	TBMW EN CMODADRS H
01	0034	00064	CEHE	M8230	CPTX	TBMN PAGE EDGE H
01	0035	00065	CEHE	M8230	CPTX	TBMW EN UNALIGN TRAP H
01	0036	00066	CEHE	M8230	CPTX	SB_M TIMEOUT TRAP L
01	0037	00067	CEHE	M8230	CPTX	SBLR RDS TRAP L
01	0038	00070	CEHE	M8230	CPTX	TBMW T& PAR UTRAP L
01	0039	00071	CEHE	M8230	CPTX	TBMW MISS UTRAP L
01	003A	00072	CEHE	M8230	CPTX	TBMW MBIT UTRAP L
01	003B	00073	CEHE	M8230	CPTX	CEHE CS PE TRAP H
01	003C	00074	CEHX	M8230	CPTX	RESERVED
01	003D	00075	CEHX	M8230	CPTX	RESERVED
01	003E	00076	CEHX	M8230	CPTX	RESERVED
01	003F	00077	CEHX	M8230	CPTX	RESERVED

ZZ-ESKAB-14.0 Documentation

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
02	0000	00000	DAPA	M8229	CPTX	IRCF OPC0 H
02	0001	00001	DAPA	M8229	CPTX	IRCF OPC1 H
02	0002	00002	DAPA	M8229	CPTX	IRCF OPC2 H
02	0003	00003	DAPA	M8229	CPTX	IRCF OPC3 H
02	0004	00004	DAPA	M8229	CPTX	IRCF OPC4 H
02	0005	00005	DAPA	M8229	CPTX	IRCF OPC5 H
02	0006	00006	DAPA	M8229	CPTX	IRCF OPC6 H
02	0007	00007	DAPA	M8229	CPTX	IRCF OPC7 H
02	0008	00010	DAPX	M8229	CPTX	CEHD MEMREF DT=B H
02	0009	00011	DAPX	M8229	CPTX	CEHD MEMREF DT=LFDG H
02	000A	00012	DAPX	M8229	CPTX	RESERVED
02	000B	00013	DAPC	M8229	CPTX	RESERVED
02	000C	00014	DAPC	M8229	CPTX	IRCE BYTF CONT H
02	000D	00015	DAPC	M8229	CPTX	IRCE WORD CONT H
02	000E	00016	DAPC	M8229	CPTX	IRCE LFDQ CONT H
02	000F	00017	DAPD	M8229	CPTX	IRCH PC REG H
02	0010	00020	DAPF	M8229	CPTX	RESERVED
02	0011	00021	DAPF	M8229	CPTX	IRCE SP1 CON2 H
02	0012	00022	DAPF	M8229	CPTX	IRCE SP1 CON1 H
02	0013	00023	DAPF	M8229	CPTX	IRCE SP1 CON0 H
02	0014	00024	DAPX	M8229	CPTX	DAPB RLOG UPDATE H
02	0015	00025	DAPD	M8229	CPTX	CEHF READ RLOG H
02	0016	00026	DAPF	M8229	CPTX	RESERVED
02	0017	00027	DPAF	M8229	CPTX	RESERVED
02	0018	00030	DAPX	M8229	CPTX	RESERVED
02	0019	00031	DAPX	M8229	CPTX	RESERVED
02	001A	00032	DAPX	M8229	CPTX	RESERVED
02	001B	00033	DAPL	M8229	CPTX	IDPN SP1 ADRO L
02	001C	00034	DAPL	M8229	CPTX	IDPN SP1 ADR1 L
02	001D	00035	DAPL	M8229	CPTX	IDPN SP1 ADR2 L
02	001E	00036	DAPL	M8229	CPTX	IDPN SP1 ADR3 L
02	001F	00037	DAPL	M8229	CPTX	IDPN SP2 ADRO L
02	0020	00040	DAPL	M8229	CPTX	IDPN SP2 ADR1 L
02	0021	00041	DAPL	M8229	CPTX	IDPN SP2 ADR2 L
02	0022	00042	DAPL	M8229	CPTX	IDPN SP2 ADR3 L
02	0023	00043	DAPL	M8229	CPTX	IDPN PRN 0 L
02	0024	00044	DAPL	M8229	CPTX	IDPN PRN 1 L
02	0025	00045	DAPL	M8229	CPTX	IDPN PRN 2 L
02	0026	00046	DAPL	M8229	CPTX	IDPN PRN 3 L
02	0027	00047	DAPX	M8229	CPTX	RESERVED
02	0028	00050	ICLT	M8231	CPTX	WCSC VCS EVEN PAR H
02	0029	00051	ICLA	M8231	CPTX	SBLM !IMO CNF INTR L
02	002A	00052	ICLA	M8231	CPTX	SBHL FAULT INTR H
02	002B	00053	ICLA	M8231	CPTX	SBHE SBI ALERT R H
02	002C	00054	ICLA	M8231	CPTX	SBLM CRD RDS INTR L
02	002D	00055	ICLA	M8231	CPTX	SBHK COMP INTR H
02	002E	00056	ICLA	M8231	CPTX	SBHE SBI REQ7 R H
02	002F	00057	ICLA	M8231	CPTX	SBHE SBI REQ6 R H
02	0030	00060	ICLA	M8231	CPTX	SBHE SBI REQ5 R H
02	0031	00061	ICLA	M8231	CPTX	SBHE SBI REQ4 R H
02	0032	00062	ICLB	M8231	CPTX	ICLB IPL ACT 4 H
02	0033	00063	ICLB	M8231	CPTX	ICLB IPL ACT 5 H
02	0034	00064	ICLB	M8231	CPTX	ICLB IPL ACT 2 H

02	0035	00065	ICLB	M8231	CPTX	ICLB IPL ACT 1 H
02	0036	00066	ICLB	M8231	CPTX	ICLB IPL ACT 0 H
02	0037	00067	ICLC	M8231	CPTX	CEHJ PRIOR 5 H
02	0038	00070	ICLC	M8231	CPTX	CEHJ PRIOR 2 H
02	0039	00071	ICLC	M8231	CPTX	CEHJ PRIOR 1 H
02	003A	00072	ICLC	M8231	CPTX	CEHJ PRIOR 0 H
02	003B	00073	ICLC	M8231	CPTX	CEHR INTR REQ L
02	003C	00074	ICLC	M8231	CPTX	CIBS CNSL RCV INTR H
02	003D	00075	ICLC	M8231	CPTX	CIBS CNSL XMIT INTR H
02	003E	00076	ICLD	M8231	CPTX	CEHC TRAP CODE2 (1) H
02	003F	00077	ICLD	M8231	CPTX	CEHC TRAP CODE1 (1) H
02	0040	00100	ICLD	M8231	CPTX	CEHC TRAP CODE0 (1) H
02	0041	00101	ICLD	M8231	CPTX	CEHP ID30 H
02	0042	00102	ICLD	M8231	CPTX	IRCE STALL+SVC L
02	0043	00103	ICLD	M8231	CPTX	CIBN HALT REQ H
02	0044	00104	ICLD	M8231	CPTX	RESERVED
02	0045	00105	ICLE	M8231	CPTX	DDPS BRANCH3 H
02	0046	00106	ICLE	M8231	CPTX	DBPV BR BIT3 H
02	0047	00107	ICLE	M8231	CPTX	DDPS BRANCH2 H
02	0048	00110	ICLE	M8231	CPTX	DBPV BR BIT2 H
02	0049	00111	ICLE	M8231	CPTX	DDPS BRANCH1 H
02	004A	00112	ICLE	M8231	CPTX	DBPV BR BIT1 H
02	004B	00113	ICLE	M8231	CPTX	DDPS BRANCH0 H
02	004C	00114	ICLE	M8231	CPTX	DBPV BR BIT0 H
02	004D	00115	ICLE	M8231	CPTX	IRCH BRC1 H
02	004E	00116	ICLE	M8231	CPTX	IRCH BRC0 H
02	004F	00117	ICLE	M8231	CPTX	IRCF OPC 0 H
02	0050	00120	ICL?	M8231	CPTX	IRCH READ OP H
02	0051	00121	ICL?	M8231	CPTX	RESERVED
02	0052	00122	ICLE	M8231	CPTX	ICLE REM BEMX S2 H
02	0053	00123	ICLE	M8231	CPTX	ICLE REM BEMX S1 H
02	0054	00124	ICLE	M8231	CPTX	ICLE REM BEMX S0 H
02	0055	00125	ICLM	M8231	CPTX	ICLM ID TO PSL H
02	0056	00126	ICLM	M8231	CPTX	ICLM ID TO VECT L
02	0057	00127	ICLM	M8231	CPTX	ICLK ID TO CES H
02	0058	00130	ICLM	M8231	CPTX	ICLM ID TO ATMP L
02	0059	00131	ICLM	M8231	CPTX	ICLM ID TO BTMP L
02	005A	00132	ICLM	M8231	CPTX	ICLM IDM S2 L
02	005B	00133	ICLM	M8231	CPTX	ICLM IDM S1 L
02	005C	00134	ICLM	M8231	CPTX	ICLM IDM S0 L
02	005D	00135	ICLJ	M8231	CPTX	DDPR SC05 (1) H
02	005E	00136	ICLJ	M8231	CPTX	DDPR SC04 (1) H
02	005F	00137	ICLJ	M8231	CPTX	DDPR SC03 (1) H
02	0060	00140	ICLJ	M8231	CPTX	DDPR SC02 (1) H
02	0061	00141	ICLJ	M8231	CPTX	DDPR SC01 (1) H
02	0062	00142	ICLJ	M8231	CPTX	DDPR SC00 (1) H
02	0063	00143	ICLJ	M8231	CPTX	ICLJ D TO ID L
02	0064	00144	ICLJ	M8231	CPTX	ICLJ ID TO Q L
02	0065	00145	ICLA	M8231	CPTX	DDPN EALU09 H
02	0066	00146	ICLK	M8231	CPTX	DDPN EALU=0 L
02	0067	00147	ICLX	M8231	CPTX	RESERVED

ZZ-ESKAB-14.0

CHAN	BIT (HEX)	Documentation BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
03	0000	00000	TBMD	M8222	CPTX	TBMD D TO MD L
03	0001	00001	TBMD	M8222	CPTX	TBMD MASK TO MD L
03	0002	00002	TBMD	M8222	CPTX	TBMD EN ID DRIVERS L
03	0003	00003	TBMS	M8222	CPTO	TBMS CPTO L
03	0004	00004	TBMF	M8222	CPTX	TBMF GRP 0 WP L
03	0005	00005	TBMF	M8222	CPTX	TBMF GRP 1 WP L
03	0006	00006	TBMC	M8222	CPTX	CAMU TB GRP 0 MATCH H
03	0007	00007	TBMC	M8222	CPTX	CAMU TB GRP 1 MATCH H
03	0008	00010	TBMU	M8222	CPTX	CEHE CMODD ADRS TRAP L
03	0009	00011	TBMU	M8222	CPTX	CEHE PAGE TRAP H
03	000A	00012	TBMX	M8222	CPTX	CEHE CS PAR ERR H
03	000B	00013	TBMX	M8222	CPTX	RESERVED
03	000C	00014	TBMN	M8222	CPTX	USCB ABORT CYCLE H
03	000D	00015	TBMN	M8222	CPTX	IRCH IB WRITE CHK H
03	000E	00016	TBMX	M8222	CPTX	RESERVED
03	000F	00017	TBMU	M8222	CPTX	TBMU CANCEL L
03	0010	00020	TBPK	M8222	CPTX	SBLB SBI PA 09 L
03	0011	00021	TBPK	M8222	CPTX	SBLB SBI PA 10 L
03	0012	00022	TBPK	M8222	CPTX	SBLB SBI PA 11 L
03	0013	00023	TBMC	M8222	CPTX	TBMC ENABLE IA H
03	0014	00024	TBMX	M8222	CPTX	RESERVED
03	0015	00025	TBMX	M8222	CPTX	RESERVED
03	0016	00026	TBMX	M8222	CPTX	SBLS IB ERR LTH H
03	0017	00027	TBMJ	M8222	CPTX	SBLT STALL L
03	0018	00030	TBMX	M8222	CPTX	RESERVED
03	0019	00031	TBMX	M8222	CPTX	RESERVED
03	001A	00032	TBMX	M8222	CPTX	RESERVED
03	001B	00033	TBMD	M8222	CPTX	CAMV MODIFY L
03	001C	00034	TBMB	M8222	CPTX	CAMV PROTECT CODE 0 L
03	001D	00035	TBMB	M8222	CPTX	CAMV PROTECT CODE 1 L
03	001E	00036	TBMB	M8222	CPTX	CAMV PROTECT CODE 2 L
03	001F	00037	TBMB	M8222	CPTX	CAMV PROTECT CODE 3 L
03	0020	00040	IDPA	M8224	CPTO	IDPA BUF B0-7(1) H
03	0021	00041	IDPA	M8224	CPTO	IDPA BUF B0-6(1) H
03	0022	00042	IDPA	M8224	CPTO	IDPA BUF B0-5(1) H
03	0023	00043	IDPA	M8224	CPTO	IDPA BUF B0-4(1) H
03	0024	00044	IDPA	M8224	CPTO	IDPA BUF B0-3(1) H
03	0025	00045	IDPA	M8224	CPTO	IDPA BUF B0-2(1) H
03	0026	00046	IDPA	M8224	CPTO	IDPA BUF B0-1(1) H
03	0027	00047	IDPA	M8224	CPTO	IDPA BUF B0-0(1) H
03	0028	00050	IDPA	M8224	CPTO	IDPA BUF B1-7(1) H
03	0029	00051	IDPA	M8224	CPTO	IDPA BUF B1-6(1) H
03	002A	00052	IDPA	M8224	CPTO	IDPA BUF B1-5(1) H
03	002B	00053	IDPA	M8224	CPTO	IDPA BUF B1-4(1) H
03	002C	00054	IDPA	M8224	CPTO	IDPA BUF B1-3(1) H
03	002D	00055	IDPA	M8224	CPTO	IDPA BUF B1-2(1) H
03	002E	00056	IDPA	M8224	CPTO	IDPA BUF B1-1(1) H
03	002F	00057	IDPA	M8224	CPTO	IDPA BUF B1-0(1) H
03	0030	00060	IDPH	M8224	CPTO	IDPH IBC 3(1) H
03	0031	00061	IDPH	M8224	CPTO	IDPH IBC 2(1) H
03	0032	00062	IDPH	M8224	CPTO	IDPH IBC 1(1) H
03	0033	00063	IDPH	M8224	CPTO	IDPH IBC 0(1) H
03	0034	00064	IDPH	M8224	CPTX	IDPH CLR 0 L

03	0035	00065	IDPH	M8224	CPTX	IRCE SAVE H
03	0036	00066	IDPA	M8224	CPTO	IDPA VAX H
03	0037	00067	IDPA	M8224	CPTX	IDPA DST R MODE H
03	0038	00070	IDPJ	M8224	CPTX	SBLR IB READ DATA L
03	0039	00071	IDPX	M8224	CPTX	RESERVED
03	003A	00072	IDPJ	M8224	CPTO	IDPJ COUNT H
03	003B	00073	IDPJ	M8224	CPTX	IDPJ FLUSH L
03	003C	00074	IDPJ	M8224	CPTX	IDPJ B5 VAL(1) H
03	003D	00075	IDPJ	M8224	CPTX	IDPJ B4 VAL(1) H
03	003E	00076	IDPJ	M8224	CPTX	IDPJ B3 VAL(0) H
03	003F	00077	IDPJ	M8224	CPTX	IDPJ B2 VAL(0) H
03	0040	00100	IDPM	M8224	CPTX	IRCD PC MODE H
03	0041	00101	IDPM	M8224	CPTX	IRCD SEL LONG L
03	0042	00102	IDPM	M8224	CPTX	IRCD SEL WORD L
03	0043	00103	IDPM	M8224	CPTX	IRCD SEL BYTE H
03	0044	00104	IDPM	M8224	CPTX	IRCE CTX 3 L
03	0045	00105	IDPM	M8224	CPTX	IRCE CTX 2 L
03	0046	00106	IDPL	M8224	CPTX	IDPL ID BUS XCVR EN L
03	0047	00107	IDPX	M8224	CPTX	RESERVED
03	0048	00110	IDPM	M8224	CPTX	IDPM B DELTA PC 2 H
03	0049	00111	IDPM	M8224	CPTX	IDPM 16 BIT B DEST L
03	004A	00112	IDPM	M8224	CPTX	IDPM B DEST H
03	004B	00113	IDPM	M8224	CPTX	IDPM B DELTA PC 1 H
03	004C	00114	IDPM	M8224	CPTX	IDPM VAXSL L
03	004D	00115	IDPM	M8224	CPTX	IDPM B DELTA PC 0 H
03	004E	00116	IDPM	M8224	CPTX	TBMX VA 01 H
03	004F	00117	IDPM	M8224	CPTX	TBMX VA 00 H
03	0050	00120	IRCH	M8223	CPTX	TBMX IB ERR L
03	0051	00121	IRCH	M8223	CPTX	TBMX TB MISS L
03	0052	00122	IRCC	M8223	CPTX	CEHH FPD BIT L
03	0053	00123	IRCX	M8223	CPTX	RESERVED
03	0054	00124	IRCE	M8223	CPTX	IRCE IB ADVANCE H
03	0055	00125	IRCJ	M8223	CPTX	IRCJ SP2 CON 1 H
03	0056	00126	IRCJ	M8223	CPTX	IRCJ SP2 CON 0 H
03	0057	00127	IRCM	M8223	CPTX	IRCM DATA EN L
03	0058	00130	IRCE	M8223	CPTO	ICLD SERVICE H
03	0059	00131	IRCE	M8223	CPTO	ICLD SERVICE BIT 0 H
03	005A	00132	IRCE	M8223	CPTO	ICLD SERVICE BIT 1 H
03	005B	00133	IRCE	M8223	CPTO	ICLD SERVICE BIT 2 H
03	005C	00134	IRCC	M8223	CPTO	IRCC EXEC CT 0 H
03	005D	00135	IRCC	M8223	CPTO	IRCC EXEC CT 1 H
03	005E	00136	IRCC	M8223	CPTO	IRCC EXEC CT 2 H
03	005F	00137	IRCX	M8223	CPTO	RESERVED

04	0000	00000	CAMP	M8220	CPTX	TBMX FORCE ERR 2 L
04	0001	00001	CAMP	M8220	CPTX	TBMX FORCE ERR 1 L
04	0002	00002	CAMP	M8220	CPTX	TBMX FORCE ERR 0 L
04	0003	00003	CAMB	M8220	CPTX	SBHF REV PAR FIELD 3 H
04	0004	00004	CAMB	M8220	CPTX	SBHF REV PAR FIELD 2 H
04	0005	00005	CAMB	M8220	CPTX	SBHF REV PAR FIELD 1 H
04	0006	00007	CAMB	M8220	CPTX	SBHF REV PAR FIELD 0 H
04	0007	00007	CAMS	M8220	CPTX	CAMS GO ADR PAR 2 OD H
04	0008	00010	CAMS	M8220	CPTX	CAMS GO ADR PAR 1 OD H
04	0009	00011	CAMS	M8220	CPTX	CAMS GO ADR PAR 0 OD H
04	000A	00012	CAMT	M8220	CPTX	CAMT G1 ADR PAR 2 OD H
04	000B	00013	CAMT	M8220	CPTX	CAMT G1 ADR PAR 1 OD H
04	000C	00014	CAMT	M8220	CPTX	CAMT G1 ADR PAR 0 OD H
04	000D	00015	CAMV	M8220	CPTX	CAMV TB PAR 2 H
04	000E	00016	CAMU	M8220	CPTX	CAMU TB PAR 1 H
04	000F	00017	CAMU	M8220	CPTX	CAMU TB PAR 0 H
04	0010	00020	CAMK	M8220	CPTX	CAMK G1 MATCH H
04	0011	00021	CAMK	M8220	CPTX	CAMK GO MATCH H
04	0012	00022	CAMP	M8220	CPTX	SBLN SBI MISS DATA G1 H
04	0013	00023	CAMP	M8220	CPTX	SBLN SBI MISS DATA G0 H
04	0014	00024	CAMM	M8220	CPT3	CAMM CPT3 B H
04	0015	00025	CAMM	M8220	CPT2	CAMM CPT2 B H
04	0016	00026	CAMM	M8220	CPT1	CAMM CPT1 B L
04	0017	00027	CAMM	M8220	CPT1	CAMM CPT1 B H
04	0018	00030	CAMP	M8220	CPTX	CAMP G1 WRITE ENABLE H
04	0019	00031	CAMP	M8220	CPTX	CAMP GO WRITE ENABLE H
04	001A	00032	CAMP	M8220	CPTX	SBHN FORCE MISS G1 H
04	001B	00033	CAMP	M8220	CPTX	SBHF FORCE MISS GO H
04	001C	00034	CAMX	M8220	CPTX	RESERVED
04	001D	00035	CAMB	M8220	CPTX	CAMB LATCH VALID BIT H
04	001E	00036	CAMX	M8220	CPTX	TBMX FORCE ERR 3 L
04	001F	00037	CAMB	M8220	CPTX	SBHF REV PAR FIELD 3 L
04	0020	00040	CAML	M8220	CPTX	CAML G1 BYTE 2 PAR OD H
04	0021	00041	CAML	M8220	CPTX	CAML G1 BYTE 2 PAR EV H
04	0022	00042	CAML	M8220	CPTX	CAML G1 BYTE 1 PAR OD H
04	0023	00043	CAML	M8220	CPTX	CAML G1 BYTE 1 PAR EV H
04	0024	00044	CAML	M8220	CPTX	CAML G1 BYTE 0 PAR OD H
04	0025	00045	CAML	M8220	CPTX	CAML G1 BYTE 0 PAR EV H
04	0026	00046	CAML	M8220	CPTX	CAML GO BYTE 2 PAR OD H
04	0027	00047	CAML	M8220	CPTX	CAML GO BYTE 2 PAR EV H
04	0028	00050	CAML	M8220	CPTX	CAML GO BYTE 1 PAR OD H
04	0029	00051	CAML	M8220	CPTX	CAML GO BYTE 1 PAR EV H
04	002A	00052	CAML	M8220	CPTX	CAML GO BYTE 0 PAR OD H
04	002B	00053	CAML	M8220	CPTX	CAML GO BYTE 0 PAR EV H
04	002C	00054	CAMB	M8220	CPTX	CAMB TAG PAR 2 EVEN H
04	002D	00055	CAMB	M8220	CPTX	CAMB TAG PAR 1 EVEN H
04	002E	00056	CAMB	M8220	CPTX	CAMB TAG PAR 0 EVEN H
04	002F	00057	CAMB	M8220	CPT1	CAMB PA LATCH 12 H
04	0030	00060	CAMB	M8220	CPT1	CAMB PA LATCH 13 H
04	0031	00061	CAMB	M8220	CPT1	CAMB PA LATCH 14 H
04	0032	00062	CAMB	M8220	CPT1	CAMB PA LATCH 15 H
04	0033	00063	CAMB	M8220	CPT1	CAMB PA LATCH 16 H
04	0034	00064	CAMB	M8220	CPT1	CAMB PA LATCH 17 H
04	0035	00065	CAMB	M8220	CPT1	CAMB PA LATCH 18 H
04	0036	00066	CAMB	M8220	CPT1	CAMB PA LATCH 19 H
04	0037	00067	CAMB	M8220	CPT1	CAMB PA LATCH 20 H

04	0038	00070	CAMB	M8220	CPT1	CAMB PA LATCH 21 H
04	0039	00071	CAMB	M8220	CPT1	CAMB PA LATCH 22 H
04	003A	00072	CAMB	M8220	CPT1	CAMB PA LATCH 23 H
04	003B	00073	CAMB	M8220	CPT1	CAMB PA LATCH 24 H
04	003C	00074	CAMB	M8220	CPT1	CAMB PA LATCH 25 H
04	003D	00075	CAMB	M8220	CPT1	CAMB PA LATCH 26 H
04	003E	00076	CAMB	M8220	CPT1	CAMB PA LATCH 27 H
04	003F	00077	CAMB	M8220	CPT1	CAMB PA LATCH 28 H
04	0040	00100	CDMX	M8221	CPTX	RESERVED
04	0041	00101	CDMX	M8221	CPTX	RESERVED
04	0042	00102	CDMX	M8221	CPTX	RESERVED
04	0043	00103	CDMU	M8221	CPT2	CDMU CPT2 H
04	0044	00104	CDMU	M8221	CPT1	RESERVED
04	0045	00105	CDMU	M8221	CPT1	RESERVED
04	0046	00106	CDMU	M8221	CPT1	CDMU CPT1 A L
04	0047	00107	CDMU	M8221	CPT1	CDMU CPT1 A H
04	0048	00110	CDMT	M8221	CPTX	TBMD EN CDM DATA L
04	0049	00111	CDMS	M8221	CPTX	CDMS G1 B3 PAR ODD H
04	004A	00112	CDMS	M8221	CPTX	CDMS G1 B3 PAR EVEN H
04	004B	00113	CDMS	M8221	CPTX	CDMS G1 B2 PAR ODD H
04	004C	00114	CDMS	M8221	CPTX	CDMS G1 B2 PAR EVEN H
04	004D	00115	CDMS	M8221	CPTX	CDMS G1 B1 PAR ODD H
04	004E	00116	CDMS	M8221	CPTX	CDMS G1 B1 PAR EVEN H
04	004F	00117	CDMS	M8221	CPTX	CDMS G1 B0 PAR ODD H
04	0050	00120	CDMS	M8221	CPTX	CDMS G1 B0 PAR EVEN H
04	0051	00121	CDMR	M8221	CPTX	CDMR G0 B3 PAR ODD H
04	0052	00122	CDMR	M8221	CPTX	CDMR G0 B3 PAR EVEN H
04	0053	00123	CDMR	M8221	CPTX	CDMR G0 B2 PAR ODD H
04	0054	00124	CDMR	M8221	CPTX	CDMR G0 B2 PAR EVEN H
04	0055	00125	CDMR	M8221	CPTX	CDMR G0 B1 PAR ODD H
04	0056	00126	CDMR	M8221	CPTX	CDMR G0 B1 PAR EVEN H
04	0057	00127	CDMR	M8221	CPTX	CDMR G0 B0 PAR ODD H
04	0058	00130	CDMR	M8221	CPTX	CDMR G0 B0 PAR EVEN H
04	0059	00131	CDMX	M8221	CPTX	RESERVED
04	005A	00132	CDMX	M8221	CPTX	RESERVED
04	005B	00133	CDMH	M8221	CPT1	CDMH ADDR LATCH 11 H
04	005C	00134	CDMH	M8221	CPT1	CDMH ADDR LATCH 10 H
04	005D	00135	CDMH	M8221	CPT1	CDMH ADDR LATCH 9 H
04	005E	00136	CDMH	M8221	CPT1	CDMH ADDR LATCH 8 H
04	005F	00137	CDMH	M8221	CPT1	CDMH ADDR LATCH 7 H
04	0060	00140	CDMH	M8221	CPT1	CDMH ADDR LATCH 6 H
04	0061	00141	CDMH	M8221	CPT1	CDMH ADDR LATCH 5 H
04	0062	00142	CDMH	M8221	CPT1	CDMH ADDR LATCH 4 H
04	0063	00143	CDMH	M8221	CPT1	CDMH ADDR LATCH 3 H
04	0064	00144	CDMH	M8221	CPT1	CDMH ADDR LATCH 2 H
04	0065	00145	CDMB	M8221	CPTX	SBHF REV PAR 3 L
04	0066	00146	CDMB	M8221	CPTX	SBHF REV PAR 2 L
04	0067	00147	CDMB	M8221	CPTX	SBHF REV PAR 1 L
04	0068	00150	CDMB	M8221	CPTX	SBHF REV PAR 0 L
04	0069	00151	CDMB	M8221	CPT3	CAMP G1 WRITE ENABLE H
04	006A	00152	CDMB	M8221	CPT3	CAMP G0 WRITE ENABLE H
04	006B	00153	CDMX	M8221	CPTX	RESERVED
04	006C	00154	CDMA	M8221	CPT2	CDMA MASK 3 H
04	006D	00155	CDMA	M8221	CPT2	CDMA MASK 2 H
04	006E	00156	CDMA	M8221	CPT2	CDMA MASK 1 H
04	006F	00157	CDMA	M8221	CPT2	CDMA MASK 0 H

05	0000	00000	SBLH	M8218	CPTX	SBHP EN ID DRIVERS L
05	0001	00001	SBLF	M8218	CPTX	SBHP ID ADDR 2 L
05	0002	00002	SBLF	M8218	CPTX	SBHP ID ADDR 1 L
05	0003	00003	SBLE	M8218	CPTX	TBMC ENABLE IA H
05	0004	00004	SBLS	M8218	CPTX	SBLS ADRS LATCH 29 H
05	0005	00005	SBLS	M8218	CPTX	IDPJ IB REQ H
05	0006	00006	SBLP	M8218	CPTX	SBLP MD TO D L
05	0007	00007	SBLF	M8218	CPTX	SBHP ID ADDR 0 L
05	0008	00010	SBLM	M8218	CPTX	SBHN CRD L
05	0009	00011	SBLP	M8218	CPTX	TBMN BUF UMCT 0 L
05	000A	00012	SBLP	M8218	CPTX	TBMN BUF UMCT 1 L
05	000B	00013	SBLP	M8218	CPTX	TBMN BUF UMCT 2 L
05	000C	00014	SBLP	M8218	CPTX	TBMN BUF UMCT 3 L
05	000D	00015	SBLP	M8218	CPTX	TBMN BUF UADS L
05	000E	00016	SBLP	M8218	CPTX	TBMN BUF UFS L
05	000F	00017	SBLM	M8218	CPTX	SPHN RDS L
05	0010	00020	SBLR	M8218	CPTX	SBHM SET INVALID L
05	0011	00021	SBLE	M8218	CPTX	SBHM SET SBI CYCLE H
05	0012	00022	SBLL	M8218	CPTX	SBHR SEND DATA H
05	0013	00023	SBLE	M8218	CPTX	SBHM ANY READ DATA L
05	0014	00024	SBLK	M8218	CPTX	SBLK LATCH TIMO REG L
05	0015	00025	SBLD	M8218	CPTX	TBMU CANCEL L
05	0016	00026	SBLW	M8218	CPTX	CLKL SYS INIT B L
05	0017	00027	SBLR	M8218	CPTX	SBLR FORCE SBI L
05	0018	00030	SBLX	M8218	CPTX	RESERVED
05	0019	00031	SBLX	M8218	CPTX	RESERVED
05	001A	00032	SBLC	M8218	CPTX	SBLC WRITE DATA 00 H
05	001B	00033	SBLC	M8218	CPTX	SBLC WRITE DATA 01 H
05	001C	00034	SBLC	M8218	CPTX	SBLC WRITE DATA 02 H
05	001D	00035	SBLC	M8218	CPTX	SBLC WRITE DATA 03 H
05	001E	C0036	SBLC	M8218	CPTX	SBLC WRITE DATA 04 H
05	001F	00037	SBLC	M8218	CPTX	SBLC WRITE DATA 05 H
05	0020	00040	SBLC	M8218	CPTX	SBLC WRITE DATA 06 H
05	0021	00041	SBLC	M8218	CPTX	SBLC WRITE DATA 07 H
05	0022	00042	SBLC	M8218	CPTX	SBLC WRITE DATA 08 H
05	0023	00043	SBLC	M8218	CPTX	SBLC WRITE DATA 09 H
05	0024	00044	SBLC	M8218	CPTX	SBLC WRITE DATA 10 H
05	0025	00045	SBLC	M8218	CPTX	SBLC WRITE DATA 11 H
05	0026	00046	SBLC	M8218	CPTX	SBLC WRITE DATA 12 H
05	0027	00047	SBLC	M8218	CPTX	SBLC WRITE DATA 13 H
05	0028	00050	SBLC	M8218	CPTX	SBLC WRITE DATA 14 H
05	0029	00051	SBLC	M8218	CPTX	SBLC WRITE DATA 15 H
05	002A	00052	SBLC	M8218	CPTX	TBMD EN SBI DATA L
05	002B	00053	SBLC	M8218	CPTX	BUS MD BYTE 0 PAR H
05	002C	00054	SBLC	M8218	CPTX	BUS MD BYTE 1 PAR H
05	002D	00055	SBLS	M8218	CPTX	SBLS SELECT SBI ADR L
05	002E	00056	SBLA	M8218	CPTX	ICLB IPL ACT 0 L
05	002F	00057	SBLA	M8218	CPTX	ICLB IPL ACT 1 L
05	0030	00060	SBHB	M8219	CPT3	SBHB WRITE DATA 16 H
05	0031	00061	SBHB	M8219	CPT3	SBHB WRITE DATA 17 H
05	0032	00062	SBHB	M8219	CPT3	SBHB WRITE DATA 18 H
05	0033	00063	SBHB	M8219	CPT3	SBHB WRITE DATA 19 H
05	0034	00064	SBHR	M8219	CPT3	SBHB WRITE DATA 20 H

ZZ-ESKAB-14.0 Documentation

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
05	0035	00065	SBHB	M8219	CPT3	SBHB WRITE DATA 21 H
05	0036	00066	SBHB	M8219	CPT3	SBHB WRITE DATA 22 H
05	0037	00067	SBHB	M8219	CPT3	SBHB WRITE DATA 23 H
05	0038	00070	SBHB	M8219	CPT3	SBHB WRITE DATA 24 H
05	0039	00071	SBHB	M8219	CPT3	SBHB WRITE DATA 25 H
05	003A	00072	SBHB	M8219	CPT3	SBHB WRITE DATA 26 H
05	003B	00073	SBHB	M8219	CPT3	SBHB WRITE DATA 27 H
05	003C	00074	SBHB	M8219	CPT3	SBHB WRITE DATA 28 H
05	003D	00075	SBHB	M8219	CPT3	SBHB WRITE DATA 29 H
05	003E	00076	SBHB	M8219	CPT3	SBHB WRITE DATA 30 H
05	003F	00077	SBHB	M8219	CPT3	SBHB WRITE DATA 31 H
05	0040	00100	SBHB	M8219	CPT1	SBHB RECEIVE MASK 0 H
05	0041	00101	SBHB	M8219	CPT1	SBHB RECEIVE MASK 1 H
05	0042	00102	SBHB	M8219	CPT1	SBHB RECEIVE MASK 2 H
05	0043	00103	SBHB	M8219	CPT1	SBHB RECEIVE MASK 3 H
05	0044	00104	SBHD	M8219	CPTX	BUS MD BYTE 2 PAR H
05	0045	00105	SBHD	M8219	CPTX	BUS MD BYTE 3 PAR H
05	0046	00106	SBHA	M8219	CPTX	SBHA BUFFER FULL L
05	0047	00107	SBHL	M8219	CPTX	SBLE LATE EXPECT RD L
05	0048	00110	SBHE	M8219	CPTX	SBLE REC PAR 0 H
05	0049	00111	SBHE	M8219	CPTX	SBLE REC PAR 1 H
05	004A	00112	SBHE	M8219	CPTX	SBLE REC PAR 2 H
05	004B	00113	SBHE	M8219	CPTX	SBLE REC PAR 3 H
05	004C	00114	SBHM	M8219	CPTX	TR SEL 1 L
05	004D	00115	SBHM	M8219	CPTX	TR SEL 2 L
05	004E	00116	SBHM	M8219	CPTX	TR SEL 4 L
05	004F	00117	SBHM	M8219	CPTX	TR SEL 8 L
05	0050	00120	SBHR	M8219	CPTX	TBMN BUF UMCT 0 L
05	0051	C0121	SBHR	M8219	CPTX	TBMN BUF UMCT 1 L
05	0052	00122	SBHR	M8219	CPTX	TBMN BUF UMCT 2 L
05	0053	00123	SBHR	M8219	CPTX	TBMN BUF UMCT 3 L
05	0054	00124	SBHR	M8219	CPTX	TBMN BUF UADS L
05	0055	00125	SBHR	M8219	CPTX	TBMN BUF UFS L
05	0056	00126	SBHM	M8219	CPT0	SBHM SELECT SBI ADRS L
05	0057	00127	SBHR	M8219	CPTX	SBHR TRANS ENABLE L
05	0058	00130	SBHD	M8219	CPTX	TBMD EN SBI DATA L
05	0059	00131	SBHE	M8219	CPTX	SBLE TRANS PAR L
05	005A	00132	SBHR	M8219	CPTX	SBLL TRANSMIT CA H
05	005B	00133	SBHM	M8219	CPTX	CEHH CUR MODE 0 H
05	005C	00134	SBHS	M8219	CPTX	CLKL SYS INIT 0 L
05	005D	00135	SBHM	M8219	CPTX	CEHH CUR MODE 1 H
05	005E	00136	SBHM	M8219	CPTX	TBMN DIG PROT L
05	005F	00137	SBHX	M8219	CPTX	RESERVED
05	0060	00140	SBHX	M8219	CPTX	RESERVED
05	0061	00141	SBHX	M8219	CPTX	RESERVED
05	0062	00142	SBHX	M8219	CPTX	RESERVFD
05	0063	00143	SBHX	M8219	CPTX	RESERVED
05	0064	00144	SBHX	M8219	CPTX	RESERVED
05	0065	00145	SBHX	M8219	CPTX	RESERVED
05	0066	00146	SBHX	M8219	CPTX	RESERVED
05	0067	00147	SBHX	M8219	CPTX	RESERVED
05	0068	00150	SBHX	M8219	CPTX	RESERVED
05	0069	00151	SBHX	M8219	CPTX	RESERVED
05	006A	00152	SBHX	M8219	CPTX	RESERVED
05	006B	00153	SBHX	M8219	CPTX	RESERVED
05	006C	00154	SBHX	M8219	CPTX	RESERVED
05	006D	00155	SBHX	M8219	CPTX	RESERVED

ZZ-ESKAB-14.0

Documentation

05 006E

00156

SBHX

M8219

CPTX

RESERVED

05 006F

00157

SBHX

M8219

CPTX

RESERVED

H 4

Fiche 1 Frame H4

Sequence 46

ZZ-ESKAB-14.0 Documentation

CHAN	BIT (HEX)	BIT (OCTAL)	DWG	MODULE	T.S.	SIGNAL NAME
06	0000	00000	FCTP	M8288	CPTX	DAPL ACC RA CONTEXT 0 H
06	0001	00001	FCTP	M8288	CPTX	DAPL ACC RA CONTEXT 1 H
06	0002	00002	FCTC	M8288	CPTX	FCTC CLR RR L
06	0003	00003	FCTH	M8288	CPTX	FCTH CP SYNC H
06	0004	00004	FNME	M8288	CPTX	FNME BUS_EXP L
06	0005	00005	FCTJ	M8288	CPTX	FCTJ ACC_N DATA H
06	0006	00006	FCTC	M8288	CPTX	FCTC ACC_Z DATA H
06	0007	00007	FCTC	M8288	CPTX	FCTC ACC_V DATA H
06	0008	00010	FNMT	M8288	CPT3	FCTD RA ADRS 3 L
06	0009	00011	FNMT	M8288	CPT3	FCTD RA ADRS 2 L
06	000A	00012	FNMT	M8288	CPT3	FCTD RA ADRS 1 L
06	000B	00013	FNMT	M8288	CPT3	FCTD RA ADRS 0 L
06	000C	00014	FNMT	M8288	CPT3	FCTP RB ADRS 3 L
06	000D	00015	FNMT	M8288	CPT3	FCTP RB ADRS 2 L
06	000E	00016	FNMT	M8288	CPT3	FCTP RB ADRS 1 L
06	000F	00017	FNMT	M8288	CPT3	FCTP RB ADRS 0 L
06	0010	00020	XXXX	M8288	CPTX	RESERVED
06	0011	00021	XXXX	M8288	CPTX	RESERVED
06	0012	00022	FNMT	M8288	CPTX	EALU C 0 L
06	0013	00023	FNMT	M8288	CPTX	FCTE COMPL L
06	0014	00024	FNMT	M8288	CPTX	FADA SPC (0) H
06	0015	00025	FNMT	M8288	CPTX	FNMS EALU CIN L
06	0016	00026	FNMT	M8288	CPTX	FCTC SEL NORM H
05	0017	00027	FNMT	M8288	CPTX	RESERVED
06	0018	00030	FNMT	M8288	CPT2	FCTN LOAD ARO H
06	0019	00031	FNMT	M8288	CPT2	FCTN LOAD AR1 H
06	001A	00032	FNMT	M8288	CPT2	FCTN LOAD ARX H
06	001B	00033	FNMT	M8288	CPT2	FCTN LOAD BR1 H
06	001C	00034	FNMT	M8288	CPT2	FCTN LOAD BRO H
06	001D	00035	FNMT	M8288	CPT0	FCTN BUS_FAD L
06	001E	00036	FNMT	M8288	CPTX	RESERVED
06	001F	00037	FNMT	M8288	CPTX	RESERVED
06	0020	00040	FNMT	M8288	CPT1	FCTN FAMX EN 0 L
06	0021	00041	FNMT	M8288	CPT3	FCTA A GT B H
06	0022	00042	FNMT	M8288	CPT3	FCTN SHF MUX EN1 L
06	0023	00043	FNMT	M8288	CPT3	FCTN SHF MUX EN0 L
06	0024	00044	FNMT	M8288	CPT1	FCTN FALU FUNC SEL 2 H
06	0025	00045	FNMT	M8288	CPT1	FCTN FALU FUNC SEL 1 H
06	0026	00046	FNMT	M8288	CPT1	FCTN FALU FUNC SEL 0 H
06	0027	00047	FNMT	M8288	CPT1	FCTN FAMX SEL 1 H
06	0028	00050	FNMT	M8288	CPT3	FCTF SHF COUNT 5 H
06	0029	00051	FNMT	M8288	CPT3	FCTF SHF COUNT 4 H
06	002A	00052	FNMT	M8288	CPT3	FCTF SHF COUNT 3 H
06	002B	00053	FNMT	M8288	CPT3	FCTF SHF COUNT 2 H
06	002C	00054	FNMT	M8288	CPT3	FCTF SHF COUNT 1 H
06	002D	00055	FNMT	M8288	CPT3	FCTF SHF COUNT 0 H
06	002E	00056	FNMT	M8288	CPT3	FCTN FALU CARRY IN H
06	002F	00057	FNMT	M8288	CPT1	FCTN FAMX SEL 0 H

J 4
29-OCT-82

Fiche 1 Frame J4

Sequence 48

3-	64	"COMMON DEFINITIONS
3-	64	"MNEUMONIC DEFINITIONS
3-	64	"GLOBAL MACRO CALLS
3-	64	"SWITCH (SWR) REGISTER BIT DEFINITIONS
3-	64	"SWITCH REGISTER 1 (SWR1) BIT DEFINITIONS
3-	64	"CONSOLE ADAPTER REGISTER DEFINITIONS
3-	64	"ID BUS REGISTER DEFINITIONS
3-	64	"LSI-11 VECTOR DEFINITIONS
3-	64	"MISCELLANEOUS DEFINITIONS
3-	64	"MODULE AND BUS NAME ASSIGNMENTS
3-	64	"LSI-11 REGISTER NAME ASSIGNMENTS
3-	64	"FILE NAME CODES
3-	64	"CONSOLE ROUTINE ERROR CODES AND DEFINITIONS
5-	166	"MICRO DIAGNOSTIC MONITOR COMMON TAGS
6-	169	"MICRO DIAGNOSTIC MONITOR INITIALIZATION
7-	270	"MICRO DIAGNOSTIC MONITOR SUBROUTINES
7-	278	"MICRO MONITOR ROUTINE
8-	340	"LOAD MONITOR ROUTINE
8-	355	"RELOAD MONITOR ROUTINE
8-	372	"LSI-11 TRAP CATCHER
9-	393	"READ A FLOPPY SECTOR ROUTINE
9-	442	"KEYBOARD INTERRUPT SERVICE ROUTINE
10-	482	"SAVE AND RESTORE REGISTERS ROUTINE
11-	511	"INTER-MONITOR FUNCTIONS
11-	526	"TRAP DISPATCHER
12-	567	"ABORT ROUTINE
12-	597	"HARDCORE DONE ROUTINE
13-	614	"MICRO TESTS (GO CHAIN) DONE ROUTINE
14-	707	"ROUTINE TO READ THE MICRO PC
15-	740	"CALL THE FAIL CHAIN MONITOR ROUTINE
16-	781	"LOAD THE WCS ROUTINE
16-	813	"LOAD ID BUS REGISTER ROUTINE
17-	845	"CALL MICRO DIAGNOSTIC MONITOR ROUTINE
17-	875	"OPEN A FILE ROUTINE
18-	931	"READ ID BUS REGISTER ROUTINE
18-	969	"READ OVERLAY ROUTINE
19-	1020	"RING THE TERMINAL BELL ROUTINE
19-	1047	"SINGLE BUS CYCLE THE CLOCK ROUTINE
20-	1067	"ENABLE CONTROL C
21-	1088	"SINGLE TIME STATE THE CLOCK ROUTINE
21-	1117	"TYPE ASCII STRING ROUTINE
22-	1245	"TYPE A 16 OR 32 BIT NUMBER ROUTINE
23-	1287	"TYPE ERROR HEADER ROUTINE
24-	1328	"TYPE FAILING MODULES ROUTINE
25-	1411	"TYPE THE SECTION NUMBER ROUTINE
25-	1441	"GLOBAL TAGS

1
2
3
4
5 .TITLE VAX 11/780 MICRO DIAGNOSTIC MONITOR
6 .IDENT /V14.0/
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
:

COPYRIGHT (C) 1977,1981
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

++
FACILITY: VAX 11/780 MICRO DIAGNOSTIC PACKAGE

FUNCTIONAL DESCRIPTION:

THIS MODULE CONTROLS THE LOADING AND EXECUTION OF THE MICRO DIAGNOSTIC HARDCORE, GO CHAIN, AND FAIL CHAIN ALONG WITH THE MICRO DIAGNOSTIC COMMAND PARSER.

ENVIRONMENT: STAND ALONE.

AUTHOR: DONALD W. MONROE, CREATION DATE: 11-OCT-1977

MODIFIED BY: DONALD W. MONROE DECEMBER 1977
DONALD W. MONROE FEBRUARY 1978
DONALD W. MONROE APRIL 1978
DONALD W. MONROE JUNE 1978
DONALD W. MONROE OCTOBER 1978
DONALD W. MONROE MARCH 1979
 THIS VERSION CONTAINS SUPPORT FOR THE MA780 MICRO
 DIAGNOSITCS.
12.0 DONALD W. MONROE JUNE 1980
 THIS VERSION SUPPORTS EUROPEAN REMOTE DIAGNOSIS
12.1 DONALD W. MONROE MARCH 1981
 HAD TO REPARTITION FLOPPY 1 AND FLOPPY 2.
13.0 DONALD W. MONROE JULY 1981
 RELEASE
13.1 BILL LANDRY/BARRY POLAND 13 DEC 1981 (SATURDAY???)
 MODIFIED TO SUPPORT REMOVAL OF WCS FILE FROM THE
 MICRO 2 FLOPPY.
14.0 DON MONROE SEPTEMBER 1981
 MODIFIED TO SUPPORT MS780-E FLOPPY
 MOVED SINGLE INSTRUCTION ROUTINE TO HARDCORE MONITOR
 AND REPLACED THE TRAP 46 CALL WITH AN ENABLE CONTROL C

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 2-1

Fiche 1 Frame L4

Sequence 50

58
59

;--

FUNCTION.

61
62
63
64 000000

.LIST MC,ME
.NLIST MD,CND
.MCALL EQUATE
EQUATE

.SBTTL "COMMON DEFINITIONS
.SBTTL " MNEMONIC DEFINITIONS

+

: FOLLOWING ARE COMMON DEFINITIONS OF MNEMONICS USED BY ALL OF THE
PROGRAMS THAT EXECUTE OUT OF THE LSI-11.

.SBTTL " GLOBAL MACRO CALLS

+

: THE FOLLOWING ".MCALL'S" ARE GLOBAL MACRO ASSIGNMENTS. THESE MACRO'S ARE
USED BY ALL 4 MONITORS, THE PARSER, AND THE DIRECTORY FILE.

: SOME OF THESE MACRO'S ARE DEFINED IN THE CONSOLE MACRO PACKAGE "STRMAC"
THEREFORE, THAT MACRO FILE MUST BE MERGED WITH THIS MACRO FILE BEFORE
ASSEMBLY.

.MCALL T\$INIT,T\$WRIT,T\$READ,F\$OPEN,F\$READ,LOADCO,CONVERT,CONABORT
.MCALL LDCNSL,GETMDM,ENCTRLC
.MCALL STARS,COMTAGS,OPENFILE,READOVR,RETURN,RESET\$,ASSEMBLE
.MCALL DONE,RDIDREG,SBCCLOCK,DONEM,FILL,CALLFAILCHAIN,\$CDDDF
.MCALL MES,TYPES,TYPED,TYPE,TYPEMOD,RINGBELL,GETUPC,TYPESECTNO
.MCALL TYPEERR,CALLMICRON,LOADWCS,STSCLOCK,LOADID,MTPS,MFPS,TYPEB

.SBTTL " SWITCH (SWR) REGISTER BIT DEFINITIONS

+

: FOLLOWING ARE THE DEFINITIONS OF THE 16 BITS OF THE SOFTWARE SWITCH
REGISTER. BITS<5:0> ARE READ WRITE BY COMMAND FROM THE MICRO MONITOR.
BITS 7 AND 6 ARE READ AND CLEAR ONLY FROM THE MICRO MONITOR.

: ALL OTHER BITS ARE TRANSPARENT TO THE OPERATOR.

000001	HALTD= 1	; HALT ON ERROR DETECTION
000002	HALTI= 2	; HALT ON ERROR ISOLATION
000004	LOOP= 4	; LOOP ON ERROR
000010	NER= 10	; NO ERROR REPORT
000020	BELL= 20	; BELL ON EVERY 6 ERRORS
000040	ERABT= 40	; GO TO NEXT TEST AFTER ERROR
000100	LOSS= 100	; LOOP ON SPECIAL SECTION
000200	LOST= 200	; LOOP ON SPECIAL TEST
000400	SINST= 400	; SINGLE INSTRUCTION FLAG
001000	FLPY2= 1000	; MA780 FLOPPY (MOUNTED) FLAG
002000	FLPY3= 2000	; FLOPPY 2 (MOUNTED) FLAG
003000	FLPY4= 3000	; MS780-E FLOPPY (MOUNTED) FLAG
003000	FLPYMSK=3000	; MASK FOR FLOPPY FIELD
004000	CONT= 4000	; CONTINUE FLAG

" SWITCH (SWR) REGISTER BIT DEFINITIONS

010000	KEYQUE= 10000	; KEYBOARD ILLEGAL CHARACTER
020000	KEYERR= 20000	; KEYBOARD ERROR FLAG
040000	CTRLC= 40000	; CONTROL C FLAG
100000	SOM= 100000	; COMMAND MODE FLAG

SBTTL " SWITCH REGISTER 1 (SWR1) BIT DEFINITIONS

FOLLOWING ARE THE BIT DEFINITIONS OF SOFTWARE SWITCH REGISTER 1 (SWR1).
THESE BITS ARE ALL TRANSPARENT TO THE OPERATOR.

000001	MARDC= 1	; (EXECUTING) FLAG
000002	RUNFLG= 2	; DIAGNOSE COMMAND HAS BEEN USED
000004	B1FULL= 4	; BUFFER 1 FULL FLAG. USED IN THE DOUBLE BUFFERED ROUTINE IN THE GO CHAIN
000010	CLKFST= 10	; SET CLOCK FAST FLAG. SET OR CLEARED BY THE OPERATOR
000020	CLKSLO= 20	; SET CLOCK SLOW FLAG.
000040	B2INUSE=40	; BUFFER 2 IN USE FLAG. USED BY THE DOUBLE BUFFER ROUTINE IN THE GO CHAIN
000100	DIRERR= 100	; DIRECTORY ERROR FLAG. SET BY THE "DIRECTORY" PROGRAM IF AN ERROR WAS DETECTED
000200	B2FULL= 200	; BUFFER 2 FULL FLAG. USED BY THE DOUBLE BUFFER ROUTINE IN THE GO CHAIN
000400	B1INUSE=400	; BUFFER 1 IN USE FLAG. USED BY THE DOUBLE BUFFER ROUTINE IN THE GO CHAIN
001000	DICMD= 1000	; DIAGNOSE COMMAND FLAG. SET WHEN A "DIAGNOSE" COMMAND IS USED
002000	MIC1FL= 2000	; GO CHAIN FILE # 1 FLAG. USED BY THE DIRECTORY SEARCH PROGRAM
004000	MIC2FL= 4000	; GO CHAIN FILE # 2 FLAG. USED BY THE DIRECTORY SEARCH PROGRAM.
010000	FPA= 10000	; FPA PRESENT FLAG. SET BY THE GO CHAIN MONITOR OR THE COMMAND PARSER.
020000	TSTSPAN=20000	; A TEST SPAN HAS BEEN SPECIFIED
040000	SCTSPAN=40000	; A SECTION SPAN HAS BEEN SPECIFIED

SBTTL " CONSOLE ADAPTER REGISTER DEFINITIONS

THE FOLLOWING ARE THE ADDRESS ASSIGNMENTS AND THE BIT DEFINITIONS OF THE CONSOLE ADAPTER REGISTERS.

173000	ROM0= 173000	; ROM LOCATION 0
173002	ROM1= 173002	; ROM LOCATION 2
173004	SPARE1= 173004	
173006	IDDATL0=173006	; LOW 16 BITS OF ID DATA REGISTER
173010	IDDATHI=173010	; HIGH 16 BITS OF ID DATA REGISTER
173012	SPARE2= 173012	
173014	RXDNE=173014	; RECEIVER CONTROL AND STATUS REGISTER
173016	TXRDY= 173016	; TRANSMITTER CONTROL AND STATUS REGISTER
173020	TOIDL0= 173020	; LO 16 BITS OF TRANSMITTER DATA BUFFER
173022	TOIDHI= 173022	; HIGH 16 BITS OF TRANSMITTER DATA BUFFER

CONSOLE ADAPTER REGISTER DEFINITIONS

173024	FMIDLO= 173024	; LO 16 BITS OF RECEIVER DATA BUFFER
173026	FMIDHI= 173026	; HIGH 16 BITS OF RECEIVER DATA BUFFER
173030	IDCS= 173030	; ID BUS CONTROL AND STATUS REGISTER
000200	IDMAINT=200	; ID MAINTENANCE BIT
000100	IDWRITE=100	; ID BUS WRITE BIT
100000	IDCYCLE=100000	; ID BUS CYCLE BIT
173032	CONNCR= 173032	; MACHINE CONTROL REGISTER
010000	INIT= 10000	CPU INITIALIZE BIT
002000	MNTRTN=2000	MAINTENANCE RETURN ENABLE BIT
001000	UPC12=1000	FORCE UPC 12 BIT
000200	CLRUWRD=200	ROM NOP BIT
000100	SOMM=100	STOP ON MICRO MATCH ENABLE BIT
000040	CLKSTPD=40	CLOCK STOPPED BIT
000020	FR1= 20	CLOCK FREQUENCY SELECT BIT 1
000010	FR0= 10	CLOCK FREQUENCY SELECT BIT 0
000004	STS= 4	ENABLE SINGLE TIME STATE BIT
000002	SBC= 2	ENABLE SINGLE BUS CYCLE BIT
000001	PROCEED=1	CLOCK PROCEED BIT
173034	CONNCS= 173034	; MACHINE CONTROL AND STATUS REGISTER
010000	FLPYON=10000	FLOPPY ON BIT
001000	CONCM=1000	CONSOLE COMMAND MODE BIT
000400	CPURUN=400	CPU RUN BIT
000200	CONACK=200	CONSOLE ACKNOWLEDGE BIT
000100	RDYIE=100	RECEIVER INTERRUPT ENABLE BIT
000040	DNEIE=40	TRANSMITTER INTERRUPT ENABLE
173036	VBCTRL= 173036	; V BUS CONTROL REGISTER
000200	CCPT0=200	TIME STATE CPT0 BIT
000100	CCPT1=100	TIME STATE CPT1 BIT
000040	CCPT2=40	TIME STATE CPT2 BIT
000020	CCPT3=20	TIME STATE CPT2 6BIT
000004	SLFTST=4	V BUS SELF TEST BIT
000002	VBLOAD=2	V BUS LOAD BIT
000001	VBCLK=1	V BUS CLOCK BIT

;SBTTL ; ID BUS REGISTER DEFINITIONS

; FOLLOWING ARE THE MNEMONICS ASSIGNED TO THE ID BUS REGISTERS.

000000	IBDAT= 00	; IBUF DATA
000001	IBTOD= 01	; TIME OF DAY CLOCK
;	;	;
000003	CONID= 03	; SYSTEM ID
000004	CONRXS= 04	; CONSOLE RXCS
000005	CONRXD= 05	; CONSOLE RXDB
000006	CONTXS= 06	; CONSOLE TXCS
000007	CONTXD= 07	; CONSOLE TXDB
000010	RWDQ= 10	; WRITE Q REG, READ D REG
000011	IBNIN= 11	; NEXT INTERVAL REGISTER
000012	IBCLKS= 12	; CLOCK CONTROL AND STATUS
000013	IBICT= 13	; IBUF INTERVAL COUNT
000014	CES= 14	
000015	VECT= 15	
000016	SIR= 16	
000017	PSL= 17	

000020 TBDFAT= 20 ; TBUF DATA
000022 TBER0= 22 ; TBUF ERROR REG 0
000023 TBER1= 23 ; TBUF ERROR REG 1
000024 ACCO= 24 ; ACCELERATOR REG 0
000025 ACC1= 25 ; ACCELERATOR REG 1
000026 ACCMNT= 26 ; ACCELERATOR MAINTENANCE REG
000027 ACCST= 27 ; ACCELERATOR STATUS REGISTER
000030 SBISILO=30 ; SBI SILO
000031 SPIERR= 31 ; SBI ERROR REG
000032 SBITO= 32 ; SBI TIMEOUT ADDRESS
000033 SBIFLT= 33 ; SBI FAULT/STATUS
000034 SBISCM= 34 ; SBI SILO COMPARATOR
000035 SBIMAT= 35 ; SBI MAINTENANCE
000036 SBICP= 36 ; SBI CACHE PARITY
000040 USCSTK=40 ; SEQUENCER MICRO STACK
000041 USCBRK=41 ; SEQUENCER MICRO BREAK
000042 USCADR=42 ; SEQUENCER WCS ADDRESS
000043 USCDAT=43 ; SEQUENCER WCS DATA
; THE FOLLOWING REGISTERS ARE THE TEMP A AND TEMP B REGISTERS
000044 POBR= 44 ;
000045 P1BR= 45 ;
000046 SBR= 46 ;
000050 KSP= 50 ;
000051 ESP= 51 ;
000052 SSP= 52 ;
000053 USP= 53 ;
000054 ISP= 54 ;
000055 FPDA= 55 ;
000056 D.SV= 56 ;
000057 Q.SV= 57 ;
000060 TEMP0= 60 ;
000061 TEMP1= 61 ;
000062 TEMP2= 62 ;
000063 TEMP3= 63 ;
000064 TEMP4= 64 ;
000065 TEMP5= 65 ;
000066 TEMP6= 66 ;
000067 TEMP7= 67 ;
000070 TEMP8= 70 ;
000071 TEMP9= 71 ;
000072 PCBB= 72 ;
000073 SCBB= 73 ;
000074 POLR= 74 ;
000075 P1LR= 75 ;
000076 SLR= 76 ;

; SBTTL " LSI-11 VECTOR DEFINITIONS

+ THE FOLLOWING MNEMONICS ARE THE DEFINITIONS FOR THE LSI-11 TRAP
AND INTERRUPT VECTORS.
-

000034
000300
000304

TRAPVEC=34 : "TRAP" INSTRUCTION VECTOR
TXVEC= 300 : TRANSMITTER INTERRUPT VECTOR
RXVEC= 304 : RECEIVER INTERRUPT VECTOR

SBTTL " MISCELLANEOUS DEFINITIONS

+ FOLLOWING ARE SOME MISCELLANEOUS DEFINITIONS USED IN THE TESTS.

177777

IDREGLO=-1

; USED AFTER A "READID" PSEUDO INSTRUCTION TO SPECIFY
; THE CONTENTS OF LOCATION "IDDATLO"

; AS THE ARGUMENT

000001

IDREGHI=1

; USED AFTER A "READID" PSEUDO INSTRUCTION
; TO SPECIFY THE CONTENTS OF LOCATION

; "IDDATHI" AS THE ARGUMENT

000034

TPCINIT=34

; FIRST ADDRESS (RELATIVE) OF EACH TEST
; STREAM OVERLAY.

; NOTE: IF THE LENGTH OF THE DISPATCH
; TABLE IS CHANGED, THIS DEFINITION

; MUST ALSO BE CHANGED.

000004

ITSTPTR=4

; FIRST ADDRESS (RELATIVE) OF THE TEST TABLE
; IN EACH TEST STREAM OVERLAY.

000010

RAD OCT= 10

; RADIX OCTAL CODE FOR CONSOLE CONVERT ROUTINE

000020

RAD HEX= 20

; RADIX HEX CODE FOR CONSOLE CONVERT ROUTINE

SBTTL " MODULE AND BUS NAME ASSIGNMENTS

+ THE FOLLOWING DEFINITIONS ARE USED BY THE "TYPMOD" ROUTINE IN THE
MICRO DIAGNOSTIC MONITOR.

000000

CIB= 0

000001

USC= 1

000002

WCS= 2

000003

PCS= 3

000004

DAP= 4

000010

DBP= 10

000005

DCP= 5

000006

DDP= 6

000007

DEP= 7

000011

CEH= 11

000012

JCL= 12

000013

CAM= 13

000014

CDM= 14

000015

TBM= 15

000016

SBL= 16

000017

SBH= 17

000020

IRC= 20

000021

IDP= 21

" MODULE AND BUS NAME ASSIGNMENTS

000022	MSB=	22
000023	MCN=	23
000024	MDT=	24
000025	MAY=	25
000026	CLK=	26
000027	TRS=	27
000030	FNH=	30
000031	FMH=	31
000032	FML=	32
000033	FAD=	33
000034	FCT=	34
000035	MAY6=	35 : MAY WITH 16K CHIP
000036	MPI=	36
000037	MPC=	37
000040	MPS=	40
000041	MAT=	41
000042	WCS2K=	42 : 2K WCS MODULE
000043	MSBE=	43 : MSB FOR MA780-E
000044	BYL=	44 : LOWER CONTROLLER
000045	BYU=	45 : UPPER CONTROLLER
000046	MAY4=	46 : 1 MEGABYTE ARRAY
000047	MAY8=	47 : 4 MEGABYTE ARRAY

: START OF BUS NAMES

000050	CSBUS=	50
000051	IDBUS=	51
000052	VBUS=	52

: START OF ADAPTER NAMES

000053	UBA=	53
000054	MBA=	54
000055	DRA=	55
000056	CIA=	56

: THE FOLLOWING OFFSET DEFINITIONS ARE OFFSETS INTO THE RAD50 LIST FOR
: THE ABOVE MODULE NAMES. IF THE LIST IS CHANGED, THESE OFFSETS MUST BE
: CHANGED. THESE OFFSETS ARE USED BY THE TYPE MODULE ROUTINE IN ESKAB.

000120	BUSOFF=	CSBUS * 2
000052	M4KOFF=	MAY * 2
000072	M6KOFF=	MAY6 * 2
000114	M64KOF=	MAY4 * 2
000116	M256KO=	MAY8 * 2
000126	ADAOFF=	UBA * 2

: SBttl " LSI-11 REGISTER NAME ASSIGNMENTS

000000	R0=	X0
000001	R1=	X1
000002	R2=	X2
000003	R3=	X3
000004	R4=	X4
000005	R5=	X5
000006	R6=	X6

000007 R7= x7
000006 SP= x6
000007 PC= x7

SBTTL " FILE NAME CODES

+
THE FOLLOWING CODES ARE USED BY THE "OPEN FILE" ROUTINE IN THE
MICRO DIAGNOSTIC MONITOR.

000000 HCMONITOR=0 : MONITOR
000002 TESTSTREAM=2 : TEST STREAM
000004 GOCHAINMONITOR=4 : GO CHAIN MONITOR
000006 GOCHA1=6 : GO CHAIN FILE NUMBER 1 (FLOPPY 1)
000010 PARSER=10 : MICRO DIAGNOSTIC PARSER
000012 GOCHA2=12 : GO CHAIN FILE NUMBER 2 (FLOPPY 2)
000014 DIRECTORY=14 : DIRECTORY SEARCH FILE
000016 FAILCHAINMONITOR=15 : FAIL CHAIN MONITOR
000020 FCHAI1=20 : FAIL CHAIN FILE NUMBER 1 (FLOPPY 1)
000022 FCHAI2=22 : FAIL CHAIN FILE NUMBER 2 (FLOPPY 2)
000024 MPGCH=24 : MA780 GO CHAIN
000026 MPFC=26 : MA780 FAIL CHAIN
000030 MSGCH=30 : MS780-E GO CHAIN
000032 MSFC=32 : MS780-E FAIL CHAIN

SBTTL " CONSOLE ROUTINE ERROR CODES AND DEFINITIONS

+
THE FOLLOWING ARE ERROR CODE DEFINITIONS AND EMT DEFINITIONS DEFINED
BY MIKE HARE THAT ARE USED TO COMMUNICATE WITH THE CONSO_C ROUTINES.

000000 \$CDDDF
000001 :FLOPPY AND TERMINAL ERROR CODES
\$FER=1 :FLOPPY HARDWARE ERROR
000002 \$FNF=2 :FILE NOT FOUND
000003 \$FNR=3 :FLOPPY QUEUE FULL
000004 \$FOR=4 :FLOPPY SECTOR # OUT OF LEGAL RANGE
000005 \$TBSY=5 :NO NODE FOR REQUEST
000006 \$TCTC=6 :CONTROL-C INPUTTED
000007 \$TER=7 :TERMINAL HARDWARE DETECTED ERROR
;USER SERVICE EMT CODE DEFINITIONS
;THESE CODES MUST BE IN SYNC WITH THE EMT SERVICE MODULE
000000 TINIT=0
000001 TWRITE=1
000002 TREAD=2
000003 OPENFL=3
000004 READSC=4
000005 WRITSC=5
000006 LOADCN=6
000007 CNVERT=?

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1.00 29-OCT-82 10:2B PAGE 3-7
" CONSOLE ROUTINE ERROR CODES AND DEFINITIONS

G 5

Fiche 1 Frame G5

Sequence 58

000010	RADGET=10
000011	OPNFL1=11
000012	TYP1=12
000013	TYP2=13
000014	LCANWC=14
000015	RMWRON=15
000016	LCWRON=16
000017	TMERTR=17
000020	R\$SET=20
000021	LDCONS=21
000022	MDMTYP=22
000023	CHKSWI=23
000024	TSTMFG=24

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 4
CONSOLE ROUTINE ERROR CODES AND DEFINITIONS

160 000000 000000*

WW=.
.BLKB 1000

H 5
Fiche 1 Frame H5
Sequence 59
: THE FIRST FOUR SECTORS (512 BYTES) OF
: THIS FILE MUST BE ZERO. THE CONSO
: THROWS THEM AWAY WHEN IT LOADS.
: START POINT FROM CONSOLE

161 000000

162

163

164 001000 000167 000446

JMP START1

166
 167 001004

.SBTTL 'MICRO DIAGNOSTIC MONITOR COMMON TAGS
 MYTAGS

:+
 : THE FOLLOWING LOCATIONS ARE USED EXCLUSIVELY BY THE MICRO DIAGNOSTIC
 : MONITOR. THEY ARE USED FOR TEMPORARY STORAGE, BUFFERS, AND TERMINAL
 : MESSAGES.
 :-

001004	000560	F2TNO: .WORD	560	: INITIAL FLPY 2 OR 3 TEST NUMBER MINUS 1
001006	000074	F2SNO: .WORD	74	: INITIAL FLPY 2 OR 3 SECTION NUMBER MINUS 1
001010	000000	LOADAD: .WORD		: BASE ADDRESS OF THIS PROGRAM (=1000)
001012	000000	STMP0: .WORD		: TEMPORARY
001014	000000	STMP1: .WORD		: ...
001016	000000	STMP2: .WORD		: COUNT FOR RING BELL ON ERROR
001020	000000	BELFLG: .WORD		: BYTE COUNT FOR READ SECTOR MACRO
001022	000000	BYTCNT: .WORD		: BYTE LENGTH FOR 'TYPEB, TYPES, AND
001024	000000	TYPLNG: .WORD	0	: TYPED' ROUTINES
001026	000000	SAVESEC: .WORD		: USED TO SAVE THE CURRENT SECTOR POINTER
001030	000000	GOSECT: .WORD		: WHEN THE MICRO DIAGNOSTIC IS INTERRUPTED
001032	001066	FILTBL: HARDCO		: USED BY THE ROUTINE THAT LOADS THE
001034	001070	HCTSTR		: FAIL CHAIN TO SAVE THE CURRENT SECTOR.
001036	001072	MICGO		: THIS TABLE POINTS AT THE RADIX 50 STRINGS
001040	001074	FLPYT1		: THAT ARE USED TO OPEN FILES ON THE FLOPPY
001042	001076	PARSE		
001044	001100	FLPYT2		
001046	001102	DIRSCH		
001050	001104	MICFAIL		
001052	001106	FCH1		
001054	001110	FCH2		
001056	001112	MPG		: MA780 GO CHAIN
001060	001114	MPF		: MA780 FAIL CHAIN
001062	001116	MSG		: MS780-E GO CHAIN
001064	001120	MSF		: MS780-E FAIL CHAIN
		UTIL		
		UTILITY=24		
001066	003270	HARDCO: .RAD50 /AC /		: THE CONSOLE, TO OPEN A FILE ON THE FLOPPY
001070	003340	HCTSTR: .RAD50 /AD /		
001072	003410	MICGO: .RAD50 /AE /		
001074	003600	FLPYT1: .RAD50 /AH /		
001076	003460	PARSE: .RAD50 /AF /		
001100	003720	FLPYT2: .RAD50 /AJ /		
001102	003530	DIRSCH: .RAD50 /AG /		
001104	003770	MICFAIL: .RAD50 /AK /		
001106	004040	FCH1: .RAD50 /AL /		
001110	004110	FCH2: .RAD50 /AM /		
001112	004160	MPG: .RAD50 /AN /		
001114	004500	MPF: .RAD50 /AP /		
001116	004420	MSG: .RAD50 /AR /		
001120	004470	MSF: .RAD50 /AS /		
001122	021103	FILBUF: .RAD50 /ESK/		
001124	000000	.WORD		: ONE OF THE ABOVE RAD50 STRINGS GETS MOVED

001126 100003

.RAD50 /TSK/

; HERE BY THE FILE OPEN ROUTINE

001130 :UTIL: .RAD50 /UTI/
001132 :.RAD50 /LIT/
001134 001 007 PERIOD: .RAD50 /TSK/
001136 COMM: MES <.> : MESSAGE FOR A DOT
ABELL: BYTE 1,7 : ASCII MESSAGE FOR ''''
TWOSPC: MES <> : ASCII MESSAGE FOR A "BELL"
001140 MSG1: MES <?UNEXPECTED TRAP TO 4...PC= >
001166 MSG2: MES <?OPEN FILE: >,NB
001200 MSG3: MES <?ERROR: >,NB
001210 MSG4: MES <TEST: >,NB
001216 MSG5: MES <FAILING MODULES: >,NB
001234 MSG7: MES <?READ SECTOR: >,NB
001250 MSG12: MES <MB2>,NB
001254 MSG13: MES <END PASS >,NB
001264 MSG14: MES <BUS>
001270 MSG24: MES < TPC= >,NB
001302 MSG25: MES <MOUNT FLOPPY ZZ-ESZAD & TYPE 'DI'>,NB
001334 MSG26: MES <SUBTEST: >,NB
001344 MSG30: MES <MB3>
001350 MSG31: MES <-L>
001354 MSG32: MES <-U>
001360 TITLE: MES <ZZ-ESKAB V14.0>,NB
001374 QUEST: MES <?>,NB

:+
: THE FOLLOWING TWO BYTES ARE USED TO SAVE THE CHARACTER THAT IS TYPED
: ON THE TERMINAL WHILE THE DIAGNOSTIC IS RUNNING. THE FIRST BYTE
: IS USED FOR THE BYTE COUNT AND THE SECOND BYTE IS USED FOR THE CHARACTER.
:-

001376
001400

KEYBUF: .BLKB 2.
TYPBUF: .BLKB 52
.EVEN

; RAD50 STRINGS GET UNPACKED HERE

238

239

240

241

242

243

244

```

245 001670 042767 003000 004530 REST3: BIC      #FLPY1,SWR      : CLEAR FLOPPY BITS
246 001676          OPENFILE MPG0CH,CHK   : IS THIS MULTIPORT FLOPPY?
247 001710 103404          BCS      1$        : BRANCH IF NO
248 001712 052767 002000 004506          BIS      #FLPY3,SWR      : SET FLOPPY 3 FLAG
249 001720 000414          BR       2$        :
250 001722 052767 001000 004476 1$:   BIS      #FLPY2,SWR      : SET FLOPPY TWO FLAG
251 001730          OPENFILE MSG0CH,CHK   : IS THIS MS780-E FLOPPY?
252 001742 103403          BCS      2$        : BRANCH IF NO
253 001744 052767 003000 004454          BIS      #FLPY4,SWR      : SET FLOPPY 4 FLAG
254 001752 032767 004000 004450 2$:   BIT      #MIC2FL,SWR1    : MIC2 SPECIAL TEST OR SECTION?
255 001760 001006          BNE      START2     : BRANCH IF YES
256 001762 016767 177016 004402          MOV      F2TNO,$TSTNM    : INIT THE FIRST TEST NUMBER
257 001770 016767 177012 004404          MOV      F2SNO,$SCTNO    : AND THE FIRST SECTION NUMBER
258
259
260
261
262
263
264
265
266
267
268

```

;+ EXECUTION RESTARTS HERE WHEN THE DIGANOSE COMMAND IS USED AFTER CHANGING
 ; FROM FLOPPY ONE TO FLOPPY 2 OR MA780 OR AUTOMATICALLY IF RUNNING ANDER "APT"
 ; AND FLOPPY ONE HAS BEEN EXECUTED.
 ;-

;+ EXECUTION IS TRANSFERED HERE (FROM ABOVE) WHENEVER THE GO CHAIN IS
 ; TO BE EXECUTED.
 ;-

```

264 001776 012737 002402' 000004 START2: MOV      #SCPUTRP,2#4    : RESTORE THE CPU TRAP VECTOR
265 002004          OPENFILE GOCHAINMONITOR : OPEN THE MICRO TESTS MONITOR FILE
266 002016 004767 000246          JSR      PC,READMON   : READ THE GO CHAIN MONITOR
267 002022 04276? 000001 004400          BIC      #HARDC,SWR1    : CLEAR HARDCORE MONITOR FLAG
268 002030 000167 004536          JMP      END+2      : GO START THE MICRO TESTS

```

```

270          SBTTL 'MICRO DIAGNOSTIC MONITOR SUBROUTINES
271          ****+
272          THE FOLLOWING ROUTINES ARE ONLY CALLED FROM THE MICRO DIAGNOSTIC MONITOR
273          (THIS PROGRAM).
274          -
275          ****+
276          SBTTL " MICRO MONITOR ROUTINE
277          ****+
278          THIS ROUTINE IS CALLED WHENEVER THE MICRO DIAGNOSTIC PARSER IS REQUIRED.
279          THE PARSER FILE IS READ INTO LSI-11 MEMORY STARTING AT THE END OF THIS
280          FILE + 512 BYTES. 512 BYTES ARE PRESERVED BECAUSE THESE BYTES CONTAIN
281          THE COMMON TAGS FOR EITHER THE HARDCORE MONITOR, THE GO CHAIN MONITOR,
282          OR THE FAIL CHAIN MONITOR. EXECUTION IS TRANSFERED TO THE PARSER
283          WITH A SUBROUTINE CALL.
284
285          WHEN THE PARSER RETURNS TO THIS ROUTINE (BY THE OPERATOR TYPING A
286          'DIAGNOSE' OR 'CONTINUE' COMMAND) A CHECK IS MADE TO SEE IF THE DIRECTORY
287          SEARCH ROUTINE IS NEEDED. IF IT IS, IT IS READ IN AND EXECUTION TRANSFERED
288          TO IT.
289
290          THEN, THE 'DIAGNOSE' FLAG IS TESTED TO SEE IF THE OPERATOR WANTED TO
291          START OVER OR CONTINUE. IF 'DIAGNOSE' WAS SPECIFIED, THE
292          STACK IS INITIALIZED AND EXECUTION TRANSFERS TO 'REST2'. IF 'CONTINUE'
293          WAS SPECIFIED, THE APPROPRIATE MONITOR (HARDCORE OR GO CHAIN) IS READ
294          BACK INTO MEMORY (EXCLUDING ITS COMMON TAGS) AND THIS ROUTINE DOES A
295          RETURN TO WHOMEVER CALLED IT.
296
297          ****+ ****+ ****+ ****+ ****+ ****+ ****+ ****+ ****+ ****+
298
299
300
301
302 002034 016767 004513 176764 MICMON: MOV SECTOR,SAVESEC : SAVE THE SECTOR OF THE CURRENTLY OPEN FILE
303 002042          4$: OPENFILE PARSER
304 002054          READOVR #TAGEND,$512.
305 002074 016700 005470          MOV TAGEND+128.,R0 : OPEN THE PARSER FILE
306 002100 162700 000600          SUB #384.,R0 : THROW AWAY THE FIRST 512 BYTES
307 002104          READOVR #TAGEND,R0 : GET THE LENGTH OF THE FILE
308 002122 004767 005242          JSR PC,TAGEND : THROW AWAY 384 BYTES
312 002126          FNCTRLC : READ THE REST OF THE FILE
316 002130 032767 001000 004272          BIT #DICMD,SWR1 : GO TO THE PARSER
317 002136 001425          BEQ 3$ : ENABLE CONTROL C'S
318 002140 032767 000300 004260          BIT #LOST+LOSS,SWR : WAS A DIAG COMMAND TYPED?
319 002146 001415          BEQ 5$ : BRANCH IF NO
320 002150          OPENFILE DIRECTORY : SPECIAL TEST OR SECTION?
321 002162 004767 000102          JSR PC,READMON : BRANCH IF NO
322 002166 004767 004400          JSR PC,END+2 : OPEN THE DIRECTORY SEARCH FILE
323 002172 032767 000100 004230          BIT #DIRERR,SWR1 : READ THE FILE
324 002200 001320          BNE 4$ : GO GET THE SECTOR NUMBER
325 002202 012706 001000          5$: MOV #1000,SP : DIRECTORY SEARCH ERROR?
326 002206 000167 177366          JMP REST2 : BRANCH IF YES
327 002212 032767 000001 004210 3$: BIT #HARDC,SWR1 : RESTORE THE STACK
328 002220 001410          BEQ 1$ : GO RESTART TESTING
329 002222          OPENFILE HCMONITOR : WAS THE HARDCORE EXECUTING?
330 002234 004767 000072          JSR PC,RELOAD : BRANCH IF NO
331 002240 000407          BR 2$ : OPEN THE HARDCORE MONITOR
332 002242          1$: OPENFILE GOCHAINMONITOR : RELOAD THE HARDCORE MONITOR
332 002242          : EXIT : OPEN THE GO CHAIN MONITOR

```

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 7-1
..
MICRO MONITOR ROUTINE

N 5

Fiche 1 Frame N5

Sequence 65

333 002254 004767 000052 JSR PC,RELOAD : RELOAD THE GO CHAIN MONITOR
334 002260 016767 170542 004270 2\$: MOV SAVESEC,SECTOR : RESTORE THE SECTOR NUMBER
335 002266 RETURN : EXIT
336
337
338

" LOAD MONITOR ROUTINE

340 .SBTTL " LOAD MONITOR ROUTINE
341 ;*****
342 :+
343 : THIS ROUTINE IS USED TO READ A MONITOR (HARDCORE, GO CHAIN, OR FAIL
344 : CHAIN) INTO LSI-11 MEMORY. IT MUST BE CALLED WITH THE FILE ALREADY OPEN
345 : I.E. LOCATION "SECTOR" MUST CONTAIN THE STARTING SECTOR NUMBER OF THE
346 : MONITOR BEING REQUESTED.
347 :
348 :*****
349
350 002270 READMON:READOVR #END,#128. : THROW AWAY FIRST 128 BYTES
351 002310 READOVR #END : READ THE REST OF THE FILE
352 002330 RETURN : EXIT
353
354
355 .SBTTL " RELOAD MONITOR ROUTINE
356 ;*****
357 :+
358 : THIS ROUTINE IS USED TO RELOAD A MONITOR (HARDCORE, OR GO CHAIN) AFTER
359 : ITS EXECUTION HAS BEEN SUSPENDED. IT PERFORMS THE SAME FUNCTION AS THE
360 : "LOAD MONITOR" ROUTINE EXCEPT THAT THE FIRST 384 BYTES OF THE FILE
361 : ARE DISCARDED. THIS PRESERVES THE STATE OF THE PARTICULAR MONITOR'S COMMON
362 : TAGS.
363 :
364 :*****
365
366 002332 RELOAD: READOVR #TAGEND,#512. : READ THE COMMON TAGS AND THROW THEM AWAY
367 002352 016700 005212 MOV TAGEND+128.,R0 : GET BYTE COUNT OF REST OF FILE
368 002356 162700 000600 SUB #384.,R0 : GET RID OF 3 MORE SECTORS
369 002362 READOVR #TAGEND,R0 : RESTORE THE FILE
370 002400 RETURN : EXIT
371
372 .SBTTL " LSI-11 TRAP CATCHER
373 ;*****
374 :+
375 : THIS ROUTINE IS POINTED TO BY LSI-11 ADDRESS 4. IT TYPES AN UNEXPECTED
376 : TRAP TO 4 MESSAGE AND RETURNS TO THE MICRO DIAGNOSTIC MONITOR.
377 :
378 :*****
379
380 002402 012667 176406 \$CPUTRP:MOV (SP)+,\$TMP1 : SAVE THE PC OF THE TRAP
381 002406 005726 TST (SP)+ : CLEANUP THE STACK
382 002410 TYPE #\$CRLF,ASCII :
383 002430 TYPE #MSG1 : TYPE THE ERROR MESSAGE
384 002446 TYPES #\$TMP1 : TYPE THE PC
385 002466 TYPE #\$CRLF,ASCII :
386 002506 1\$: CALLMICMON : GO TO THE MICRO MONITOR
387 002510 000776 BR 1\$: DON'T ALLOW CONTINUE
388
389
390
391

" READ A FLOPPY SECTOR ROUTINE

```

393      .SBTTL  "      READ A FLOPPY SECTOR ROUTINE
394
395
396      :+
397      : THIS SUBROUTINE IS CALLED BY THE "READ OVERLAY" ROUTINE.
398      : IT READS N SECTORS FROM THE FLOPPY STARTING AT THE SECTOR SPECIFIED BY
399      : THE CONTENTS OF LOCATION "SECTOR". THE NUMBER OF SECTORS TO READ (N)
400      : IS AT 2(SP) WHEN THIS ROUTINE IS ENTERED. THE ADDRESS OF THE BUFFER
401      : TO PUT THE DATA INTO IS AT (SP) WHEN THE ROUTINE IS ENTERED.
402
403      : IF THE C BIT (IN THE PSW ON THE STACK AT 12(SP) ) IS SET A REQUEST IS
404      : MADE TO THE FLOPPY DRIVER TO READ N SECTORS AND INTERRUPT WHEN
405      : COMPLETE. IF THE C BIT IS CLEAR, THE FLOPPY DRIVER DOES NOT RETURN UNTIL
406      : THE DATA HAS BEEN READ IN.
407
408      : IF A FLOPPY ERROR OCCURS DURING THE READ, AN ERROR MESSAGE IS TYPED
409      : ALONG WITH THE ERROR CODE, AND THE CONSOLE IS REBOOTED.
410
411      : THE CONTENTS OF "SECTOR" IS ADVANCED THE NUMBER OF SECTORS JUST READ.
412
413
414

```

415 002512 016667 000002 176276	READDISK:	MOV 2(SP),\$TMP2	: GET THE NUMBER OF SECTORS
416 002520 012616		MOV (SP)+,(SP)	: CLEANUP THE STACK
417 002522 011667 176264		MOV (SP),\$TMP0	: GET THE BUFFER ADDRESS
418 002526 032766 000001 000010		BIT \$1,10(SP)	: IS THE C BIT SET ON THE STACK?
419 002534 001424		BEC 2\$: BRANCH IF NO
423 002536		F\$READ SECTOR,\$TMP0,F\$PYVEC,\$TMP2	: READ N SECTORS INTERRUPT DRIVEN
424 002604 000421		BR 3\$: CONTINUE
425 002606		2\$: F\$READ SECTOR,\$TMP0,,\$TMP2	: READ N SECTORS
429 002650 103032		3\$: BCC 1\$: BRANCH IF NO ERRORS
430 002652 012667 176136		MOV (SP)+,\$TMP1	: GET THE ERROR CODE
431 002656		TYPE #SCRLF,ASCII	: TYPE ERROR MESSAGE
432 002676		TYPE #MSG7	: TYPE THE ERROR CODE
433 002714		TYPES #\$TMP1	: RETURN TO CONSOLE
434 002734		CONABORT	: UPDATE SEC. OR POINTER
435 002736 066767 176054 003612	1\$:	ADD \$TMP2,SECTOR	: RETURN
436 002744 000200		RTS R0	

```

440
441
442      .SBTTL  "      KEYBOARD INTERRUPT SERVICE ROUTINE
443
444
445      :+
446      : THIS ROUTINE HANDLES KEYBOARD INTERRUPTS. IF A CTRL C IS TYPED
447      : THE CONTROL C FLAG IS SET IN THE SWITCH REGISTER. IF ANY OTHER CHARACTER
448      : IS TYPED, A QUESTION MARK IS ECHOED AND EXECUTION CONTINUES.
449
450      : IF AN ERROR IS DETECTED, AN ERROR MESSAGE IS TYPED AND THE ERROR
451      : CODE AND THE CONSOLE IS REBOOTED.
452
453

```

454 002746 103053	KEYINT: BCC 3\$: BRANCH IF NO ERROR
455 002750 026627 000002 000006	CMP 2(SP),#\$TCTC	: CTRL C?
456 002756 001004	BNE 1\$: BRANCH IF NO
457 002760 052767 040000 003440	BIS #CTRLC,SWR	: SET CONTROL C FLAG IN SWR
458 002766 000443	BR 3\$	

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 9-1
" KEYBOARD INTERRUPT SERVICE ROUTINE

D 6
459 002770 1\$: TYPE #\$CRLF,ASCII
460 003010 TYPE #MSGC
461 003026 016667 000002 003506 MOV 2(SP),KEYCODE ; TYPE THE ERROR MESSAGE
462 003034 TYPES #KEYCODE ; GET THE ERROR CODE
463 003054 TYPE #\$CRLF,ASCII ; TYPE IT
464 003074 CONABORT ; RETURN TO CONSOLE
465
469 003076 3\$: ENCTRLC ; ENABLE CONTROL C'S
473 003100 RETURN ;
474
475
476 003102 TYPPASS:TYPE #\$CRLF,ASCII
477 003122 TYPE #MSG13
478 003140 TYPES #SPASS,HEX
479 003160 TYPE #\$CRLF,ASCII
480 003200 RETURN

Fiche 1 Frame D6

Sequence 68


```

511          .SBTTL "INTER-MONITOR FUNCTIONS
512          ****+*****
513
514          THE FOLLOWING ROUTINES ARE USED BY ALL THE MONITORS (MICRO DIAGNOSTIC,
515          HARDCORE, GO CHAIN, AND FAIL CHAIN). THEY ARE CALLED BY EXECUTING A
516          "TRAP" INSTRUCTION WITH THE APPROPRIATE CODE IN THE THE LOW BYTE OF THE
517          INSTRUCTION. THE CALLS TO THESE ROUTINES ARE DEFINED BY MACRO'S SINCE
518          MOST OF THE ROUTINES REQUIRE THAT ARGUMENTS FROM THE CALLING PROGRAM
519          BE LOCATED IN SPECIFIC PLACES, IN THE "GLOBAL" TAGS BLOCK.
520
521          THE DESCRIPTION OF EACH ROUTINE DESCRIBES WHICH GLOBAL TAGS ARE USED
522          AND HOW THEY ARE INTERPRETED.
523
524          ****+*****
525
526          .SBTTL " TRAP DISPATCHER
527          ****+*****
528
529          THE "TRAP" INSTRUCTION VECTOR POINTS TO THIS ROUTINE. THIS ROUTINE
530          TAKES THE LOW BYTE OF THE TRAP INSTRUCTION, GENERATES AN INDEX, AND
531          DISPATCHES TO THE APPROPRIATE ROUTINE.
532
533          ****+*****
534
535 003246 010046      $TRAP:   MOV     R0,-(SP)      : SAVE R0
536 003250 016600 000002      MOV     2(SP),R0      : GET THE PC OF THE CALL
537 003254 005740      TST     -(R0)      : BACK IT UP TO THE TRAP INSTRUCTION
538 003256 111000      MOVB    (R0),R0      : GET THE TRAP NUMBER
539 003260 016000 003266'      MOV     $STRPAD(R0),R0      : GET THE ADDRESS OF THE ROUTINE
540 003264 000200      RTS     R0      : DISPATCH TO THE ROUTINE
541
542 003266 003342'      $TRPAD: ABORT      : ABORT TO THE CONSOLE
543 003270 004214'      $OPNFILE      : OPEN FILE ROUTINE
544 003272 004456'      $READOV      : READ OVERLAY ROUTINE
545 003274 005272'      $TYPES       : TYPE 16 BIT NUMBER ROUTINE
546 003276 005302'      $TYPED       : TYPE 32 BIT NUMBER ROUTINE
547 003300 004724'      $TYPE        : TYPE ASCII STRING ROUTINE
548 003302 005666'      $TYPMOD      : TYPE LIST OF MODULES ROUTINE
549 003304 004554'      $RNGBEL      : RING THE BELL ROUTINE
550 003306 005372'      $TYPERR      : TYPE ERROR HEADER ROUTINE
551 003310 004200'      $MICMON      : CALL MICRO MONITOR ROUTINE
552 003312 003722'      $FAILCHAIN      : CALL FAIL CHAIN ROUTINE
553 003314 004044'      $LDWCS       : LOAD THE WCS ROUTINE
554 003316 004672'      $STSCLK      : SINGLE TIME STATE THE CLOCK ROUTINE
555 003320 003370'      $DONE        : DONE WITH HARDCORE
556 003322 004132'      $LOADID      : LOAD ID BUS REGISTER ROUTINE
557 003324 004360'      $RDIDRE      : READ ID BUS REGISTER ROUTINE
558 003326 005262'      $TYPEB       : TYPE AN 8 BIT NUMBER
559 003330 004624'      $SBCCLO      : SINGLE BUS CYCLE THE CLOCK ROUTINE
560 003332 003376'      $DONEM       : DONE WITH MICRO TESTS ROUTINE
561 003334 004650'      $ENCTRLC      : ENABLE CONTROL C
562
563 003336 003630'      : $INST        : SINGLE INSTRUCTION THE HARDCORE ROUTINE
564 003340 006270'      : $GETUPC      : READ THE UPCSV REG FROM THE V BUS
565                      : TYPSEC       : ROUTINE TO TYPE THE CURRENT SECTION NUMBER
566                      : $UTILITY      : FAILCHAIN DEBUG UTILITY PROGRAM

```

567 .SBTTL " ABORT ROUTINE
568 :*****
569 :+
570 : THIS ROUTINE IS CALLED WITH A "TRAP 0" INSTRUCTION.
571 : THE ROUTINE INITIALIZES THE VAX 11/780 CPU AND REBOOTS THE CONSOLE.
572 : IT IS USED WHEN THE MICRO DIAGNOSTICS ARE COMPLETE OR IF A FATAL
573 : LSI-11 ERROR OCCURS.
574 :*****
575 :
576 :
577 :*****
578 579 003342 012737 010001 173032 ABORT: MOV #INIT+PROCEED,@#CONMCR ; START THE CLOCK AND INIT
580 003350 005037 173032 CLR @#CONMCR ; CLEAR THE INIT
584 003354 005005 CLR R5 ; INIT R5 FOR CONSOLES PRIOR TO VERSION 05-00
585 003356 GETMDM ; GET THE CONSOLE TYPE
586 003360 005705 TST R5 ; US OR CCITT VERSION?
587 003362 001401 BEQ 1\$; BRANCH IF US VERSION
588 003364 LDCNSL ; RELOAD FOR CCITT VERSION
589 003366 1\$: LOADCON ; RELOAD FOR US VERSION
593
594
595
596
597 .SBTTL " HARDCORE DONE ROUTINE
598 :*****
599 :+
600 : THIS ROUTINE IS CALLED WITH A "TRAP 32" INSTRUCTION.
601 : IT TRANSFERS CONTROL TO "START2" WHICH STARTS EXECUTION
602 : OF THE GO CHAIN.
603 :
604 : IT IS USED BY THE HARDCORE MONITOR WHEN THE HARDCORE IS COMPLETE.
605 :*****
606 :
607 :
608 609 003370 022626 \$DONE: CMP (SP)+, (SP)+ ; CLEANUP THE STACK
610 003372 000167 176400 JMP START2 ; GO TO MICROTTESTS
611
612

" MICRO TESTS (GO CHAIN) DONE ROUTINE

```

614          .SBTLL  "      MICRO TESTS (GO CHAIN) DONE ROUTINE
615          ;*****
616          ;+
617          ; THIS ROUTINE IS CALLED BY A "TRAP 44" INSTRUCTION.
618
619          ; IT CHECKS THE PASS COUNT TO SEE IF THE OPERATOR REQUESTED MORE THAN
620          ; ONE PASS TO BE EXECUTED. IF ANOTHER PASS IS TO BE EXECUTED, IT PASSES
621          ; CONTROL TO 'REST' WHICH STARTS EXECUTING THIS FLOPPY AGAIN.
622
623          ; IF THE REQUIRED NUMBER OF PASSES IS COMPLETE, IT CHECKS TO SEE IF
624          ; THIS IS FLOPPY NUMBER 2 OR 3. IF IT IS NOT (THIS IS FLOPPY 1) IT TRIES TO
625          ; OPEN THE FLOPPY 2 GO CHAIN FILE. IF THIS FAILS, IT TRIES TO OPEN THE MA780
626          ; GO CHAIN. IF THIS FAILS, A MESSAGE IS TYPED TO
627          ; MOUNT THE SECOND FLOPPY. THEN THE PARSER IS
628          ; READ IN AND CONTROL PASSES TO IT.
629
630          ; IF THIS IS FLOPPY NUMBER 2 OR 3, EXECUTION TRANSFERS TO THE ABORT ROUTINE.
631
632          ; THIS ROUTINE IS CALLED BY THE GO CHAIN MONITOR WHEN IT REACHES THE
633          ; END OF THE CURRENT GO CHAIN FILE.
634
635          ;*****
636
637 003376 012706 001000      $DONEM: MOV    #1000,SP      ; INIT THE SP
638 003402 042767 006000 003020   BIC    #MIC1FL+MIC2FL,SWR1 ; CLEAR LOOP FLAGS
639 003410 005267 002754      INC    SPASS      ; INCREMENT THE PASS COUNT
640 003414 032767 003000 003004   BIT    #FLPYMSK,SWR   ; IS THIS FLOPPY 2 OR 3 OR 4?
641 003422 001031              BNE    2$          ; BRANCH IF YES
642
643          ;+
644          ; THIS IS FLOPPY 1, CHECK IF RUNNING ON APT SYSTEM
645
646
647 003424      OPENFILE GOCHA2,CHK      ; IS FLOPPY 2 GO CHAIN HERE?
648 003436 103006      BCC    10$          ; BRANCH IF YES
649 003440      OPENFILE MPGOCH,CHK      ; IS MA780 GO CHAIN HERE?
650 003452 103405      BCS    1$          ; BRANCH IF NO
651
652          ;+
653          ; FLOPPY 1, APT SYSTEM
654
655
656 003454 052767 000002 002746  10$: BIS    #RUNFLG,SWR1
657 003462 000167 176202      JMP    REST3        ; START EXECUTION OF FLOPPY 2
658
659          ;+
660          ; FLOPPY 1, NOT APT
661
662
663 003466 004767 177410      1$:  JSR    PC,TYPPASS
664 003472 026767 002672 003045   CMP    SPASS,PASCNT ; TYPE THE CURRENT PASS COUNT
665 003500 103035      BHIS   6$          ; DONE ALL THE SPECIFIED PASSES?
666 003502 000167 176110      JMP    REST          ; BRANCH IF YES
667          ; RESTART FLOPPY 1
668
669          ;+
670          ; FLOPPY 2 OR 3
671

```

671
672 003506 2\$: OPENFILE GOCHA1,CHK ; ON THE APT SYSTEM?
673 003520 103414 BCS 4\$; BRANCH IF NO
674 003522 004767 177354 JSR PC,TYPPASS ; TYPE THE CURRENT PASS COUNT
675 003526 026767 002636 003012 CMP \$PASS,PASCNT ; DONE ALL PASSES?
676 003534 103401 BLO 3\$; BRANCH IF NO
677 003536 CONABORT ; RETURN TO CONSOLE
678
679 ;+
680 ; FLOPPY 2, APT, MORE PASSES
681 ;-
682
683 003540 042767 003000 002660 3\$: BIC #FLPYMSK,SWR ; RESTART WITH FLOPPY 1
684 003546 000167 176044 JMP REST
685
686 ;+
687 ; FLOPPY 2, NOT APT
688 ;-
689
690 003552 004767 177324 4\$: JSR PC,TYPPASS ; TYPE THE PASS COUNT
691 003556 026767 002606 002762 CMP \$PASS,PASCNT ; DONE ALL PASSES?
692 003564 103401 BLO 5\$; BRANCH IF NO
693 003566 000411 BR 7\$; GO TO PARSER
694 003570 000167 176022 5\$: JMP REST ; RESTART FLOPPY 2
695
696 ;+
697 ; FLOPPY 1, NO APT, DONE ALL PASSES
698 ;-
699
700 003574 6\$: TYPE #MSG25 ; TYPE "MOUNT FLOPPY 2" MESSAGE
701 003612 005067 002552 7\$: CLR \$PASS ; INIT THE PASS COUNT
702 003616 042767 000002 002604 BIC #RUNFLG,SWR1 ; CLEAR THE RUN FLAG, SO "CONT" CAN'T BE TYPED
703 003624 000167 176204 JMP MICMON ; GO TO PARSER
704
705

" ROUTINE TO READ THE MICRO PC

707 SBTTL " ROUTINE TO READ THE MICRO PC
708 :*****
709 :+
710 : THIS ROUTINE IS CALLED WITH A "TRAP 50" INSTRUCTION.
711 :
712 : THIS ROUTINE READS THE V BUS AND ASSEMBLES BITS <12:0> OF CHANNEL
713 : ZERO (CONTENTS OF THE UPC SAVE REGISTER) INTO BITS <12:0> OF LOCATION
714 : "GOTUPC" IN THE GLOBAL TAGS AREA.
715 :
716 : IT IS CALLED WHENEVER THE CURRENT MICRO PC IS REQUIRED.
717 :
718 :*****
719 :
720 003630 004767 177346 \$GETUPC:JSR PC_SAVER ; SAVE THE REGISTERS
721 003634 012700 000015 MOV #13.,R0 ; SET THE LOOP COUNT
722 003640 005002 CLR R2 ; INITIALIZE R2
723 003642 ^52737 000002 173036 BIS #VBLOAD,A#VBCTRL ; LOAD THE V BUS
724 003650 042737 000002 173036 BIC #VBLOAD,A#VBCTRL ;
725 003656 113701 173037 1\$: MOVB A#VBCTRL+1,R1 ; GET A BIT OF THE UPC
726 003662 052737 000001 173036 BIS #VBCLK,A#VBCTRL ; CLOCK THE VBUS
727 003670 006001 ROR R1 ; PUT THE UPC BIT IN R2
728 003672 006002 ROR R2 ;
729 003674 005300 DEC R0 ; DONE WITH 13 BITS?
730 003676 001367 BNE 1\$; BRANCH IF NO
731 003700 000241 CLC ;
732 003702 006002 ROR R2 ; ADJUST THE RESULT
733 003704 006202 ASR R2 ;
734 003706 006202 ASR R2 ;
735 003710 010267 002554 MOV R2,GOTUPC ; PUT RESULT IN UPC LOCATION
736 003714 004767 177304 JSR PC,RESTR ; RESTORE THE REGISTERS
737 003720 000002 RTI ; RETURN
738

740 .SBTTL " CALL THE FAIL CHAIN MONITOR ROUTINE
741 :*****
742 :+
743 : THIS ROUTINE IS USED TO LOAD AND EXECUTE THE FAIL CHAIN MONITOR.
744 :*****
745 :
746 :
747 003722 \$FAILCHAIN:
748 003722 000240 NOP ; \REPLACE WITH RTI TO REMOVE
749 003724 016767 002626 175076 MOV SECTOR,GOSECT ; SAVE THE CURRENT SECTOR NUMBER
750 003732 OPENFILE FAILCHAINMONITOR ; OPEN THE FAIL CHAIN MONITOR FILE
751 003744 READOVR #TAGEND,#512. ; GET THE FIRST 512 BYTES
752 003764 016700 003600 MOV TAGEND+128.,R0 ; GET BYTE COUNT OF FILE
753 003770 162700 000600 SUB #384.,R0 ; THROW AWAY FIRST 384 BYTES
754 003774 READOVR #TAGEND,RO ; READ THE GO CHAIN MONITOR
755 004012 004767 003352 JSR PC,TAGEND ; EXECUTE THE FAIL CHAIN
756 004016 OPENFILE GOCHAINMONITOR ; RESTORE THE GO CHAIN MONITOR
757 004030 004767 176276 JSR PC,RELOAD ;
758 004034 016767 174770 002514 MOV GOSECT,SECTOR ; RESTORE THE SECTOR NUMBER
759 004042 000002 RTI ; RETURN TO THE GO CHAIN
760 :
761 :
762 :\$UTILITY:
763 : RTI ; \REPLACE WITH RTI TO REMOVE
764 : MOV SECTOR,GOSECT
765 : OPENFILE UTILITY
766 : READOVR #TAGEND,#512.
767 : MOV TAGEND+128.,R0
768 : SUB #384.,R0
769 : READOVR #TAGEND,RO
770 : JSR PC,TAGEND
771 : OPENFILE PARSER
772 : READOVR #TAGEND,#512.
773 : MOV TAGEND+128.,R0
774 : SUB #384.,R0
775 : READOVR #TAGEND,RO
776 : MOV GOSECT,SECTOR
777 : MOV #TAGEND+4,(SP) ; SETUP THE RETURN PC
778 : ADD TAGEND+2,(SP)
77. : RTI

781 SBTTL " LOAD THE WCS ROUTINE
782 *****
783 :+
784 : THIS ROUTINE IS CALLED WITH A "TRAP 26" INSTRUCTION.
785 :
786 : FOLLOWING ARE THE REQUIREMENTS FOR THE GLOBAL LOCATIONS:
787 :
788 : "WCSADR" - CONTAINS THE ADDRESS OF THE WCS LOCATION TO LOAD
789 : "SRCADR" - CONTAINS THE ADDRESS OF THE DATA TO LOAD INTO THE LOCATION
790 : "WCSCNT" - CONTAINS THE NUMBER OF 96 BIT WORDS TO LOAD TIMES 3.
791 :
792 : THIS ROUTINE LOADS THE DATA POINTED TO BY "SRCADR" INTO WCS STARTING AT
793 : THE LOCATION POINTED TO BE "WCSADR". "WCSCNT" NUMBER OF 32 BIT WORDS
794 : IS LOADED.
795 :
796 : THIS ROUTINE IS CALLED WHENEVER A MICRO WORD(S) ARE REQUIRED TO BE LOADED
797 : INTO THE WCS.

801 004044				SLDWCS: LOADID	#WCSADR, #USCADR	LOAD THE WCS ADDRESS INTO THE ADDRESS REGISTER
802 004070				1\$: LOADID	SRCADR, #USCDAT	LOAD THE DATA INTO THE WCS
803 004114	062767	000004	002326	ADD	#4, SRCADR	INCREMENT THE ADDRESS OF THE DATA TO THE
804						NEXT 32 BIT WORD
805 004122	005367	002326		DEC	WCSCNT	DONE LOADING?
806 004126	001360			BNE	1\$	BRANCH IF NO
807 004130	000002			RTI		RETURN

SBTTL " LOAD ID BUS REGISTER ROUTINE

+ THIS ROUTINE IS CALLED WITH A "TRAP 34" INSTRUCTION.
THE GLOBAL LOCATIONS USED BY THIS ROUTINE ARE DEFINED AS FOLLOWS:
 "IDADR" - CONTAINS THE ADDRESS (IN BITS<5:0>) OF THE ID BUS
 REGISTER TO BE LOADED.
 "IDDAT" - CONTAINS THE ADDRESS OF THE DATA TO LOAD INTO THE REGISTER.
THIS ROUTINE TRANSFERS THE DATA POINTED TO BY "IDDAT" INTO THE ID BUS
REGISTER SPECIFIED BY "IDADR".
THIS ROUTINE IS CALLED WHENEVER AN ID BUS REGISTER IS REQUIRED TO BE
LOADED WITH DATA.

```

832 004132 116737 002324 173030 $LOADID:MOV B IDADR, @#IDCS ; PUT THE REGISTER NUMBER IN THE CONTROL REGISTER
833 004140 052737 000300 173030 BIS #IDMAINT+IDWRITE, @#IDCS ;
834 004146 010046 MOV RO,-(SP) : SAVE RO
835 004150 016700 002304 MOV IDDAT,RO : GET THE ADDRESS OF THE DATA
836 004154 012037 173020 MOV (RO)+, @#TOIDLO : LOAD THE "TO ID REGISTER" LOW 16 BITS
837 004160 011037 173022 MOV (RO), @#TOIDHI : LOAD THE HI 16 BITS

```

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 16-1
" LOAD ID BUS REGISTER ROUTINE

M 6

Fiche 1 Frame M6

Sequence 77

838 004164
839 004166 042737 000100 173030 SBC CLOCK
840 004174 012600 BIC #IDWRITE,2#IDCS ; TICK THE CLOCK
841 004176 000002 MOV (SP)+,R0 ; SET BACK TO READ
 RTI ; RESTORE R0
842
843 ; RETURN

845 .SBTTL " CALL MICRO DIAGNOSTIC MONITOR ROUTINE
846 :*****
847 :+
848 : THIS ROUTINE IS CALLED WITH A "TRAP 22" INSTRUCTION.
849 : IT INITIALIZES THE TERMINAL AND CALLS THE MICRO MONITOR
850 : SUBROUTINE. UPON RETURN, IT REQUESTS AN INTERRUPT DRIVEN READ FROM THE
851 : KEYBOARD.
852 : IT IS USED WHENEVER THE PARSER IS REQUIRED. (TYPICALLY AFTER ERROR
853 : REPORTS OR WHEN THE OPERATOR TYPES A CTRL C)
854 :
855 :*****
856 :
857 :*****
858 :
862 004200 004767 175626 SMICMON:TSINIT : CLEAR ANY READ FUNCTIONS
863 004202 : JSR PC,MICMON : GO TO THE MONITOR
864 (04206 TSINIT : CLEAR ANY READ FUNCTIONS
865 004210 ENCTRLC : ENABLE CONTROL C'S
869 004212 000002 RTI : RETURN
870
871
872
873
874
875 .SBTTL " OPEN A FILE ROUTINE
876 :*****
877 :+
878 : THIS ROUTINE IS CALLED WITH A "TRAP 2" INSTRUCTION.
879 : IT USES THE FOLLOWING GLOBAL LOCATIONS:
880 :
881 : "FILPTR" - CONTAINS THE INDEX INTO THE FILE NAME TABLE.
882 : THE FOLLOWING GLOBAL LOCATIONS ARE LOADED BY THIS ROUTINE:
883 : "SECTOR" - CONTAINS THE STARTING SECTOR NUMBER OF THE FILE.
884 :
885 : THIS ROUTINE GENERATES A POINTER TO THE FILE NAME, BASED ON THE CONTENTS
886 : OF "FILPTR", AND MAKES A CALL TO THE CONSOLE TO OPEN THE FILE. IF THE
887 : OPEN IS SUCCESSFUL, THE CONSOLE RETURNS THE STARTING SECTOR NUMBER AND
888 : THE LENGTH OF THE FILE WHICH ARE SAVED IN LOCATIONS "SECTOR" AND
889 : "ENDSECT". IF THERE IS ANY FLOPPY ERROR, EXCEPT "FILE NOT FOUND", THE
890 : CONSOLE IS REBOOTTED. IF A "FILE NOT FOUND" ERROR OCCURS AND THE
891 : C BIT (IN THE PSW THAT WAS PUSHED ON THE STACK) WAS SET WHEN THIS ROUTINE
892 : WAS CALLED, A RETURN TO THE CALLING MONITOR IS MADE WITH THE C BIT SET.
893 : IF THE C BIT WAS NOT SET, THE CONSOLE IS REBOOTTED.
894 :
895 : IF THE OPEN IS SUCCESSFUL, THE C BIT UPON RETURN TO THE CALLING MONITOR,
896 : IS CLEAR.
897 :
898 : THIS ROUTINE IS USED WHENEVER A NEW FILE IS REQUIRED FROM THE FLOPPY.
899 :
900 :
901 :
902 :
903 004214 062767 001032' 002214 SOPNFIL:AND #FILBL,FILPTR : GENERATE ADDRESS OF ADR OF RAD50
904 004222 017767 002210 17'674 MOV #FILPTR,FILBUF+2
905 004230 017767 174670 174666 MOV #FILBUF+2,FILBUF+2 :...
909 004236 FSOPEN #FILBUF : OPEN THE FILE
913 004244 103036 BCC 1\$: BRANCH IF NO ERROR

" OPEN A FILE ROUTINE

914 004246 012667 174542	MOV (SP)+,\$TMP1	; GET THE ERROR CODE
915 004252 022767 000002 174534	CMP #\$FNF,\$TMP1	; FILE NOT FOUND?
916 004260 001010	BNE 2\$; BRANCH IF NO
917 004262 032766 000001 000002	BIT #1,2(SP)	; IS C BIT SET?
918 004270 001404	BEQ 2\$; BRANCH IF NO
919 004272 052766 000001 000002	BIS #1,2(SP)	; SET THE C BIT ON THE STACK
920 004300 000426	BR 3\$; RETURN
921 004302	2\$: TYPE #MSG2	; TYPE THE ERROR MESSAGE
922 004320	TYPES #\$TMP1	; TYPE THE ERROR CODE
923 004340	CONABORT	; RETURN TO CONSOLE
924 004342 012667 002210	1\$: MOV (SP)+,SECTOR	; GET THE START SECTOR OF THIS FILE
925 004346 005726	TST (SP)+	; THROW AWAY LENGTH OF FILE
926 004350 042766 000001 000002	BIC #1,2(SP)	; CLEAR THE RETURN C BIT
927 004356 000002	3\$: RTI	; RETURN
928		
929		

931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952

SBTTL " READ ID BUS REGISTER ROUTINE

+ THIS ROUTINE IS CALLED WITH A "TRAP 36" INSTRUCTION.
THE FOLLOWING GLOBAL LOCATIONS ARE REQUIRED:
"IDADR" - CONTAINS THE ADDRESS (IN BITS <5:0>) OF THE ID BUS
REGISTER THAT IS TO BE READ.
THE FOLLOWING LOCATIONS ARE LOADED BY THIS ROUTINE:
"RDIDLO" - CONTAINS THE LOW 16 BITS OF THE ID BUS REGISTER READ.
"RDIDHI" - CONTAINS THE HIGH 16 BITS OF THE ID BUS REGISTER READ.
THIS ROUTINE READS THE ID BUS REGISTER SPECIFIED BY "IDADR" INTO LOCATIONS
"RDIDLO" AND "RDIDHI".
THIS ROUTINE IS USED WHENEVER THE CONTENTS OF AN ID BUS REGISTER
ARE REQUIRED.
-

953 004360 \$RDIDRE:

954 004360 116737 002076 173030 MOVB IDADR, #IDCS ; LOAD THE REGISTER NUMBER
955 004366 052737 000200 173030 BIS #IDMAINT, #IDCS ; SET THE ID MASTER BIT
956 004374 STSCLOCK #2 ; TICK THE CLOCK
957 004404 042737 000200 173030 BIC #IDMAINT, #IDCS
958 004412 STSCLOCK #1 ;
959 004422 052737 000200 173030 BIS #IDMAINT, #IDCS ;
960 004430 013737 173006 002026 MOV #IDDATLO, RDIDLO ; PUT THE LOW WORD ON THE STACK
961 004436 013767 173010 002022 MOV #IDDATHI, RDIDHI ; PUT THE HIGH WORD ON THE STACK
962 004444 STSCLOCK #1 ; RETURN CLOCK TO CPTO
963 004454 000002 RTI ; RETURN
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979

SBTTL " READ OVERLAY ROUTINE

+ THIS ROUTINE IS CALLED BY A "TRAP 4" INSTRUCTION.
THE FOLLOWING GLOBAL LOCATIONS ARE REQUIRED:
"OVRADR" - CONTAINS THE ADDRESS OF THE BUFFER TO READ
THE OVERLAY INTO.
"OVRBYT" - CONTAINS THE NUMBER OF BYTES IN THE OVERLAY.
THIS ROUTINE READS AN OVERLAY OF THE CURRENTLY OPEN FILE. THE FIRST
SECTOR READ IS THE CONTENTS OF "SECTOR". IF THE CONTENTS OF "OVRBYT"
IS EQUAL TO A MINUS 1, THE NUMBER OF BYTES TO READ COMES FROM THE FIRST
WORD OF THE FIRST SECTOR READ OTHERWISE, THE CONTENTS OF "OVRBYT"
IS THE NUMBER OF BYTES READ.
THE ROUTINE TAKES THE BYTE COUNT (EITHER FROM "OVRBYT" OR BY READING THE
THE FIRST SECTOR AND GETTING THE FIRST WORD) AND CALCULATES THE NUMBER

" READ OVERLAY ROUTINE

988 ; OF SECTORS THAT IS REQUIRED TO READ THAT NUMBER OF BYTES. A CALL IS THEN
 989 ; MADE TO THE "READ DISK" ROUTINE TO ACTUALLY READ THE DATA.
 990 ;
 991 ; THIS ROUTINE IS USED WHENEVER DATA IS REQUIRED TO BE READ OFF THE FLOPPY.
 992 ;-
 993 ;*****

95 .04456 010046	\$READOV:MOV R0,-(SP)	; SAVE R0
996 004460 010146	MOV R1,-(SP)	; AND R1
997 004462 016700 001752	MOV OVRADR,R0	; GET ADR TO READ OVERLAY
998 004466 016701 001750	MOV OVRBYT,R1	; GET BYTE COUNT
999 004472 100011	BPL 2\$; BRANCH IF BYTE COUNT IS IN R1
1000 004474 012746 000001	MOV #1,-(SP)	; PUT SECTOR COUNT ON THE STACK
1001 004500 004067 176006	JSR R0,READDISK	; READ A SECTOR (128. BYTES)
1002 004504 011001	MOV (R0),R1	; GET THE BYTE COUNT FROM THE DATA JUST READ
1003 004506 062700 000200	ADD #128.,R0	; UPDATE BUFFER POINTER
1004 004512 162701 000200	SUB #128.,R1	; DECREMENT THE BYTE COUNT
1005 004516 000301	SWAB R1	; PUT # OF SECTORS IN LOW BYTE
1006 004520 006301	ASL R1	
1007 004522 005501	ADC R1	
1008 004524 032701 177000	1\$: BIT #177000,R1	; WAS THERE A FRACTION OF A SECTOR?
1009 004530 001401	BEQ 3\$; BRANCH IF NO
1010 004532 005201	INC R1	; MUST READ 1 MORE SECTOR
1011 004534 042701 177000	3\$: BIC #177000,R1	; CLEAR UPPER 7 BITS (FOR APT DRIVER)
1012 004540 010146	MOV R1,-(SP)	; PUT SECTOR COUNT ON STACK
1013 004542 004067 175744	JSR R0,READDISK	; READ N SECTORS
1014 004546 012601	MOV (SP)+,R1	; RESTORE R1
1015 004550 012600	MOV (SP)+,R0	; RESTORE R0
1016 004552 000002	RTI	; EXIT
1017		
1018		

" RING THE TERMINAL BELL ROUTINE

1020 .SBTTL " RING THE TERMINAL BELL ROUTINE
1021 :*****
1022 :+
1023 : THIS ROUTINE IS CALLED WITH A "TRAP 16" INSTRUCTION.
1024 :
1025 : THIS ROUTINE SENDS A BELL TO THE TERMINAL ON THE FIRST CALL AND THEN
1026 : ON EVERY SIXTH CALL. THIS IS DONE SO THAT INTERMITTENT FAILURES ARE
1027 : EASIER TO DETECT.
1028 :
1029 : THIS ROUTINE IS USED BY THE HARDCORE MONITOR WHEN THE "BELL ON ERROR"
1030 : FLAG IS SET.
1031 :
1032 :*****

1033 004554 005767 174240 \$RNGBEL:TST BELFLG : FIRST CALL?
1034 004560 001406 174230 BEQ 1\$: BRANCH IF YES
1035 004562 022767 000006 174230 CMP #6,BELFLG : 6 CALLS YET?
1036 004570 003012 BGT 2\$: BRANCH IF NO
1037 004572 005067 174222 CLR BELFLG : INITIALIZE THE FLAG
1038 004576 1\$: TYPE #ABELL,ASCII : TYPE THE BELL
1039 004616 005267 174176 2\$: INC BELFLG : COUNT THIS CALL
1040 004622 000002 RTI : RETURN
1041
1042
1043
1044
1045
1046

1047 .SBTTL " SINGLE BUS CYCLE THE CLOCK ROUTINE
1048 :*****
1049 :+
1050 : THIS ROUTINE IS CALLED WITH A "TRAP 42" INSTRUCTION.
1051 :
1052 : THIS ROUTINE CLEARS THE STS BIT, SETS THE SBC BIT, AND SETS THE PROCEED
1053 : BIT IN THE "MACHINE CONTROL REGISTER (MCR)". THE EFFECT OF THIS SEQUENCE
1054 : IS TO ADVANCE THE CPU CLOCK TO THE NEXT CPT0.
1055 :
1056 : THIS ROUTINE IS USED WHENEVER THE CPU CLOCK IS REQUIRED TO BE TICKED
1057 : AND TERMINATED IN CPT0.
1058 :
1059 :*****

1060 1061 004624 052737 000002 173032 \$SBCCLO:BIS #SBC,@#CONMCR ; ENSURE SINGLE BUS CYCLE SET
1062 004632 042737 000004 173032 BIC #STS,@#CONMCR ; ENSURE STS CLEAR
1063 004640 052737 000001 173032 BIS #PROCEED,@#CONMCR ; TICK THE CLOCK
1064 004646 000002 RTI
1065

1067 .SBTTL " ENABLE CONTROL C
1068 :*****
1069 :+
1070 : THIS ROUTINE IS CALLED BY A "TRAP 46" INSTRUCTION.
1071 :
1072 : THIS ROUTINE QUEUES A KEYBOARD REQUEST FOR ONE CHARACTER SO THAT
1073 : CONTROL C'S CAN BE DETECTED.
1074 :
1075 :-----
1076 :*****
1077 004650
1081 004650
1085 004670 000002
1086 \$ENCTRLC:
 T\$READ #KEYBUF,#1,#KEYINT
 RTI

" SINGLE TIME STATE THE CLOCK ROUTINE

1088 SBTTL " SINGLE TIME STATE THE CLK. < ROUTINE
1089 :*****
1090 :+
1091 : THIS ROUTINE IS CALLED BY A "TRAP 30" INSTRUCTION.
1092 :
1093 : THE FOLLOWING GLOBAL VARIABLE IS REQUIRED:
1094 :
1095 : "STSNO" - CONTAINS THE NUMBER OF TIME STATES REQUIRED.
1096 :
1097 : THIS ROUTINE SETS THE STS AND SBC BITS IN THE MCR REGISTER AND THEN
1098 : SETS THE PROCEED BIT THE NUMBER OF TIMES SPECIFIED BY THE CONTENTS
1099 : OF "STSNO". THE ROUTINE EXITS WITH THE STS BIT CLEAR.
1100 :
1101 : THIS ROUTINE IS USED ANYWHERE THE CPU CLOCK IS REQUIRED TO BE ADVANCE
1102 : LESS THAN ONE BUS CYCLE.
1103 :
1104 :*****
1105
1106 004672 052737 000006 173032 \$STSCLK:BIS #STS+SBC,@#CONMCR ; SET THE SINGLE TIME STATE BIT
1107 004700 052737 000001 173032 1\$: BIS #PROCEED,@#CONMCR ; TICK THE CLOCK
1108 004706 005367 001544 DEC STSNO ; DONE YET?
1109 004712 001372 BNE 1\$; BRANCH IF NO
1110 004714 042737 000004 173032 BIC #STS,@#CONMCR ; RESET THE STS BIT
1111 004722 000002 RTI ; RETURN
1112
1113
1114
1115
1116

1117 SBTTL " TYPE ASCII STRING ROUTINE
1118 :*****
1119 :+
1120 : THIS ROUTINE IS CALLED BY A "TRAP 12" INSTRUCTION.
1121 :
1122 : THE FOLLOWING GLOBAL VARIABLES ARE REQUIRED:
1123 :
1124 : "TYPADR" - CONTAINS THE START ADDRESS OF THE ASCII STRING TO TYPE.
1125 :
1126 : THE FOLLOWING GLOBAL LOCATION IS LOADED BY THIS ROUTINE:
1127 :
1128 : "BYTCNT" - CONTAINS THE NUMBER OF BYTES IN THE ASCII STRING.
1129 :
1130 : IF THE C BIT IS SET ON HE STACK, THE STRING POINTED TO IS CONVERTED FROM
1131 : PADIX 50 TO ASCII BEFORE TYPING. THE ASCII BUFFER STARTS AT LOCATION 320.
1132 :
1133 : THIS ROUTINE PICKSUP THE FIRST BYTE OF THE STRING POINTED TO BY "TYPADR"
1134 : AT SAVES IT IN 'BYTCNT'. THEREFORE, THE FIRST BYTE OF THE STRING PASSED
1135 : TO THIS ROUTINE MUST BE THE BYTE COUNT OF THE STRING. A CALL IS THEN
1136 : MADE TO THE TERMINAL DRIVER TO TYPE THE STRING.
1137 :
1138 : IF AN ERROR IS DETECTED DURING THE TYPING OF THE STRING (TERMINAL ERROR)
1139 : THE CONSOLE PROGRAM IS REBOOTTED.
1140 :
1141 : THIS ROUTINE IS USED ANYTIME AN ASCII STRING IS TO BE TYPED ON THE TERMINAL.
1142 :
1143 :*****
1144

```

1145 004724 032766 000001 000002 $TYPE: BIT #1,2(SP) ; IS C BIT SET?
1146 004732 001402 BEQ 12$ ; BRANCH IF NO
1147 004734 000167 000260 JMP 15$ ; STRING IS ALREADY ASCII
1148
1149
1150 ;+
1151 ; THE FOLLOWING ROUTINE CONVERTS A RADIX 50 STRING TO ASCII
1152 ; NOTE: THE STRING MUST BE GENERATED BY THE 'MES' MACRO
1153 ;-
1154 004740 004767 176236 12$: JSR PC.SAVER ; SAVE THE REGISTERS
1155 004744 016704 001474 MOV TYPADR,R4 ; GET POINTER TO RAD50 STRING
1156 004750 012705 001400' MOV #TYPBUF,R5 ; SET THE BUFFER ADDRESS
1157 004754 012701 000050 MOV #50,R1 ; SET THE DIVISOR
1158 004760 012767 000001 MOV #1,$TMPO ; SET A FLAG
1159
1160 004766 012400 174024 3$: MOV (R4)+,R0 ; GET RAD50 WORD
1161 004770 012702 177777 MOV #-1,R2 ; INIT QUOTIENT
1162 004774 005202 1$: INC R2 ; START DIVIDE BY 50(0)
1163 004776 160100 SUB R1,R0 ; ...
1164 005000 103375 BCC 1$ ; ...
1165 005002 060100 ADD R1,R0 ; R0 CONTAINS 3RD CHARACTER
1166 005004 110065 000002 MOVB R0,2(R5) ; PUT IN BUFFER
1167
1168 005010 012700 177777 2$: MOV #-1,R0 ; INIT QUOTIENT
1169 005014 005200 INC R0 ; START DIVIDE BY 50(0)
1170 005016 160102 SUB R1,R2 ; ...
1171 005020 103375 BCC 2$ ; ...
1172 005022 060102 ADD R1,R2 ; R2 CONTAINS 2ND CHARACTER
1173 005024 110265 000001 MOVB R2,1(R5) ; PUT IN BUFFER
1174 005030 110015 MOVB R0,(R5) ; PUT 1ST CHARACTER IN BUFFER
1175
1176 005032 005367 173754 DEC $TMPO ; DO WE HAVE BYTE COUNT YET?
1177 005036 100402 BMI 11$ ; BRANCH IF YES
1178 005040 110003 MOVB R0,R3 ; PUT BYTE COUNT IN R3
1179 005042 005203 INC R3 ; ADJUST FOR BYTE COUNT CHARACTER
1180
1181 005044 062705 000003 11$: ADD #3,R5 ; UPDATE BUFFER POINTER
1182 005050 162703 000003 SUB #3,R3 ; DONE YET?
1183 005054 003344 BGT 3$ ; BRANCH IF NO
1184
1185 ; CONVERT BUFFER TO ASCII
1186
1187 005056 012700 001400' MOV #TYPBUF,R0 ; GET THE BUFFER POINTER
1188 005062 112001 MOVB (R0)+,R1 ; GET THE RAD50 BYTE COUNT
1189 005064 010003 MOV R0,R3 ; SETUP THE DESTINATION POINTER
1190 005066 010104 MOV R1,R4 ; INIT THE ASCII BYTE COUNT
1191 005070 105710 4$: TSTB (R0) ; CHARACTER = 0?
1192 005072 001003 BNE 5$ ; BRANCH IF NO
1193 005074 112723 000040 MOVB #40,(R3)+ ; MAKE IT ASCII 'SPACE'
1194 005100 000435 BR 10$ ; ...
1195 005102 122710 000033 5$: CMPB #33,(R0) ; CHARACTER =33 OR LESS THAN?
1196 005106 002414 BLT 7$ ; BRANCH IF NO
1197 005110 001404 BEQ 6$ ; BRANCH IF =33
1198 005112 111013 MOVB (R0),(R3) ; MOVE THE CHARACTER
1199 005114 152723 000100 BISB #100,(R3)+ ; MAKE ASCII ALPHA
1200 005120 000425 BR 10$ ; ...
1201 005122 005200 6$: INC R0 ; POINT AT REAL CHARACTER

```

```

1202 005124 111013      MOVB   (R0), (R3)      ; OVERWRITE FLAG CHARACTER
1203 005126 005304      DEC    R4          ; DECREASE THE ASCII BYTE COUNT
1204 005130 005301      DEC    R1          ; AND THE RAD50 BYTE COUNT
1205 005132 152723 000040  BISB   #40, (R3)+ ; CONVERT TO ASCII
1206 005136 000416      BR    10$        ;
1207 005140 122710 000035 7$:    CMPB   #35, (R0)  ; CHARACTER = 34 OR 35?
1208 005144 002407      BLT    9$        ; BRANCH IF NO
1209 005146 001403      BEQ    8$        ; BRANCH IF =35
1210 005150 112723 000056  MOVB   #56, (R3)+ ; MAKE IT ASCII '..'
1211 005154 00040?      BR    10$        ;
1212 005156 112723 000050 8$:    MOVB   #50, (R3)+ ; MAKE IT ASCII "("
1213 005162 000404      BR    10$        ;
1214 005164 111002      9$:    MOVB   (R0), R2      ; GET THE CHARACTER
1215 005166 062702 000022  ADD    #22, R2      ; MAKE IT ASCII NUMERIC
1216 005172 110223      MOVB   R2, (R3)+ ; PUT BACK IN BUFFER
1217 005174 005200      10$:   INC    R0          ; UPDATE SOURCE POINTER
1218 005176 005301      DEC    R1          ; DONE YET?
1219 005200 001333      BNE    4$        ; BRANCH IF NO
1220 005202 11046? 174172  MOVB   R4, TYPBUF ; UPDATE ASCII BYTE COUNT
1221 005206 004767 176012  JSR    PC, RESTR ; RESTORE THE REGISTERS
1222
1223
1224 ;+
1225 ; CONVERSION IS COMPLETE
1226 ;-
1227 005212 01276? 001400' 001224 15$:   MOV    #TYPBUF, TYPADR ; SET NEW BUFFER POINTER
1228 005220 11776? 001220 173574  MOVB  @TYPADR, BYTCNT ; GET THE BYTE COUNT OF THE STRING
1229 005226 00526? 001212 13$:   INC    TYPADR      ; POINT THE POINTER AT THE START OF THE STRING
1230 005232 103004      TSWRIT TYPADR, BYTCNT ; GO TYPE THE STRING
1231 005246 02262? 000007  BCC    14$        ; BRANCH IF NO ERROR
1232 005250 001366      CMP    (SP)+, #$TER ; IS IT A FATAL ERROR?
1233 005254 000002      BNE    13$        ; BRANCH IF NO
1234 005256            CONABORT      ; RETURN TO CONSOLE
1235 005260            RTI          ; RETURN
1236
1237

```

" TYPE A 16 OR 32 BIT NUMBER ROUTINE

```

1245 .SBTTL " TYPE A 16 OR 32 BIT NUMBER ROUTINE
1246 ;*****
1247 ;+
1248 ; THIS ROUTINE IS CALLED BY EITHER A "TRAP 6" (TO TYPE A 16 BIT NUMBER)
1249 ; OR BY A "TRAP 10" (TO TYPE A 32 BIT NUMBER) OR A "TRAP 40" (TO
1250 ; TYPE AN 8 BIT NUMBER) INSTRUCTION.
1251
1252 ; THIS ROUTINE REQUIRES THE FOLLOWING GLOBAL VARIABLE:
1253
1254 ; "TYPADP" - CONTAINS THE ADDRESS OF THE NUMBER TO TYPE.
1255
1256 ; THIS ROUTINE TAKES THE CONTENTS OF THE LOCATION POINTED TO BY "TYPADR"
1257 ; (AND THE LOCATION + 2 FOR 32 BITS), CONVERTS THE DATA TO AN ASCII
1258 ; STRING, AND CALLS THE "TYPE" ROUTINE TO TYPE THE STRING.
1259
1260 ; IF THE C BIT (IN THE PSW ON THE STACK) IS SET THE DATA IS CONVERTED TO
1261 ; HEXADECIMAL OTHERWISE, THE DATA IS CONVERTED TO OCTAL. LEADING ZERO'S
1262 ; IN THE DATA ARE NOT TRUNCATED.
1263
1264 ; THIS ROUTINE IS USED ANYTIME A NUMBER IS TO BE TYPED ON THE TERMINAL.
1265
1266 ;*****
1267
1268 005262 012767 000001 173534 $TYPEB: MOV #1,TYPLNG ; SET THE BYTE LENGTH
1269 005270 000407 BR TYPESD
1270 005272 012767 000002 173524 $TYPES: MOV #2,TYPLNG ; SET THE BYTE LENGTH
1271 005300 000403 BR TYPESD
1272 005302 012767 000004 173514 $TYPED: MOV #4,TYPLNG ; SET THE BYTE LENGTH
1273 005310 004767 175666 TYPESD: JSR PC,SAVER ; SAVE THE REGISTERS
1274 005314 016700 001124 MOV TYPADR,R0 ; GET ADDRESS OF NUMBER TO TYPE
1275 005320 012702 000020 MOV #RADHEX,R2 ; INIT R2
1276 005324 032766 000001 000020 BIT #1,20(SP) ; IS C BIT SET ON THE STACK?
1277 005332 001002 BNE 1$ ; BRANCH IF YES
1278 005334 012702 000010 MOV #RADOCT,R2 ; SET RADIX TO OCTAL
1279 005340 016701 173460 1$: MOV TYPLNG,R1 ; GET THE BYTE LENGTH
1280 005344 005344 104007 2$: CONVERT ; CONVERT THE DATA TO ASCII
1281 005346 EMT CNVERT
1282 005364 004767 175634 TYPE R0,ASCII ; TYPE THE DATA
1283 005370 000002 JSR PC,RESTR ; RESTORE THE REGISTERS
1284
1285 RTI

```

```

1287      .SBTTL  "      TYPE ERROR HEADER ROUTINE
1288      :*****+
1289      :+      THIS ROUTINE IS CALLED BY A "TRAP 20" INSTRUCTION.
1290      :+      THIS ROUTINE REQUIRES THE FOLLOWING GLOBAL VARIABLES:
1291      :+      '$ERRPC' - CONTAINS THE ERROR PC TO BE TYPED.
1292      :+      '$STSTNM' - CONTAINS THE TEST NUMBER TO BE TYPED.
1293      :+      '$SUBTST' - CONTAINS THE SUBTEST NUMBER TO BE TYPED.
1294      :+      THIS ROUTINE TYPES AN ERROR HEADER OF THE FOLLOWING FORMAT:
1295      :+      ?ERROR: <$ERRPC> TEST: <$STSTNM> SUBTEST: <$SUBTST>
1296      :+      THIS ROUTINE IS USED BY THE HARDCORE AND GO CHAIN MONITORS TO
1297      :+      TYPE MICRO DIAGNOSTIC ERROR MESSAGES.
1298      :+      :*****+
1299      :+      :*****+
1300      :+      :*****+
1301      :+      :*****+
1302      :+      :*****+
1303      :+      :*****+
1304      :+      :*****+
1305      :+      :*****+
1306      :+      :*****+
1307 005372      $TYPERR:TYPE      #$CRLF,ASCII
1308 005412      TYPE      #MSG3      : TYPE 'ERROR: '
1309 005430 032766 000001 000002      BIT      #1,2(SP)      : IS C BIT SET?
1310 005436 001411      BEQ      1$      : BRANCH IF NO
1311 005440      TYPES      #$ERRPC,HEX      : TYPE IN HEX
1312 005460 000410      BR      2$      :
1313 005462      1$:      TYPES      #$ERRPC      : TYPE THE ERROR NUMBER
1314 005502      2$:      TYPE      #TWOSPC      : TYPE TWO SPACES
1315 005520      TYPE      #MSG4      : TYPE 'TEST: '
1316 005536      TYPES      #$STSTNM,HEX      : TYPE THE TEST NUMBER
1317 005556      TYPE      #TWOSPC      :
1318 005574      TYPE      #MSG26      : TYPE "SUBTEST"
1319 005612 005767 000562      TST      SUBTST      : IS THE SUBTEST NUMBER 0?
1320 005616 001002      BNE      3$      : BRANCH IF NO
1321 005620 005267 000554      INC      SUBTST      : MAKE IT 1
1322 005624      3$:      TYPES      #$SUBTST,HEX      : TYPE THE SUBTEST NUMBER
1323 005644      TYPE      #$CRLF,ASCII      :
1324 005664 000002      RTI      : RETURN
1325
1326

```


ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 24-1
" TYPE FAILING MODULES ROUTINE

M 7

Fiche 1 Frame M7

Sequence 90

1385 006132 000421			BR	51\$	
1386 006134 122710 000044	6\$:	CMPB	#BYL,(R0)	: LOWER CONTROLLER?	
1387 006140 001003		BNE	52\$: BRANCH IF NO	
1388 006142 012703 001350'		MOV	#MSG31,R3	: TYPE "-L"	
1389 006146 000405		BR	53\$		
1390 006150 122710 000045	52\$:	CMPB	#BYU,(R0)	: UPPER CONTROLLER	
1391 006154 001010		BNE	51\$		
1392 006156 012703 001354'		MOV	#MSG32,R3	: TYPE "-U"	
1393 006162	53\$:	TYPE	R3	: TYPE THE SUFFIX	
1394 006176	51\$:	TYPE	#COMSPC	: TYPE A ", "	
1395 006214 105720		TSTB	(R0)+	: POINT TO NEXT MODULE OFFSET	
1396 006216 000646		BR	1\$: SEE IF THERE IS MORE MODULES	
1397					
1398 006220 162701 000053	5\$:	SUB	#UBA,R1	: GET INDEX OF ADAPTERS (STARTS AT UBA)	
1399 006224 006301		ASL	R1	: MAKE IT A 32 BIT INDEX	
1400 006226 006301		ASL	R1		
1401 006230 066701 000330		ADD	MODLNK,R1	: GET ADDRESS OF NAMES OF ADAPTERS	
1402 006234 062701 000126		ADD	#ADAOFF,R1		
1403 006240 000672		BR	4\$: GO TYPE THE ADAPTER NAME	
1404					
1405 006242	2\$:	TYPE	#SCRLF,ASCII		
1406 006262 004767 174736		JSR	PC,RESTR	: RESTORE THE REGISTERS	
1407 006266 000002		RTI		: EXIT	
1408					
1409					

1411 .SBTTL " TYPE THE SECTION NUMBER ROUTINE
1412 :*****
1413 :+
1414 : THIS ROUTINE IS CALLED BY A "TRAP 52" INSTRUCTION.
1415 :
1416 : THIS ROUTINE REQUIRES THE FOLLOWING GLOBAL VARIABLE:
1417 :
1418 : "SSCTNO" - CONTAINS THE SECTION NUMBER CURRENTLY LOADED.
1419 :
1420 : THIS ROUTINE CONVERTS THE CONTENTS OF "SSCTNO" TO A 2 DIGIT HEXADECIMAL
1421 : STRING AND TYPES THE STRING FOLLOWED BY A COMMA. IF THE CONTENTS OF
1422 : SSCTNO IS EQUAL TO 18(X) OR 2F(X), A CARRIAGE RETURN LINE FEED PRECEDES
1423 : THE ABOVE TYPEOUT.
1424 :
1425 : THIS ROUTINE IS USED BY THE HARDCORE AND GO CHAIN MONITORS TO TYPE THE
1426 : NUMBER OF THE SECTION JUST LOADED.
1427 :
1428 :*****
1429 :
1430 006270 122767 000030 000104 TYPSEC: CMPB #30,\$SSCTNO : TIME TO START A NEW LINE?
1431 006276 001404 BEQ 2\$: BRANCH IF YES
1432 006300 122767 000057 000074 CMPB #57,\$SSCTNO : TIME TO START THE THIRD LINE?
1433 006306 001010 BNE 1\$: BRANCH IF NO
1434 006310 2\$: TYPE #SCRLF,ASCII :
1435 006330 1\$: TYPEB #SSCTNO,HEX : TYPE THE SECTION NUMBER
1436 006350 2\$: TYPE #COMMA :
1437 006366 000C02 RTI : EXIT
1438 :
1439 :
1440 :
1441 006370 COMTAGS
.SBTTL "GLOBAL TAGS
:*****
: THE FOLLOWING 128 BYTES ARE THE GLOBAL TAGS USED BY ALL THE MONITORS.
: THESE TAGS MUST BE LOCATED AT THE END OF THE MICRO DIAGNOSTIC MONITOR
: AND AT THE BEGINNING OF ALL THE OTHER MONITORS OR FILES THAT USE THESE
: TAGS.
: ONCE THE MICRO DIAGNOSTIC MONITOR IS LOADED INTO MEMORY, THESE TAGS ARE
: NEVER OVERLAYERED.
: THESE TAGS MUST BE EXACTILY 128 BYTES IN LENGTH.

006370 000000 \$PASS: .WORD 0 : CONTAINS THE CURRENT PASS COUNT
006372 000000 STSTNM: .WORD 0 : CONTAINS THE CURRENT TEST NUMBER
006374 000000 ENDSNP: .WORD 0 : ENDING TEST OR SECTION NUMBER OF SPAN
006376 000000 TESTNO: .WORD 0 : CONTAINS THE TEST NUMBER FOR LOST
006400 000000 SUBTST: .WORD 0 : CONTAINS THE CURRENT SUBTEST NUMBER
006402 000000 SSCTNO: .WORD 0 : CONTAINS THE CURRENT SECTION NUMBER
006404 000001 SECTNO: .WORD 1 : CONTAINS THE SECTION NUMBER FOR LOSS
006406 000000 SERFLG: .WORD 0 : IS NON ZERO IF AN ERROR HAS BEEN DETECTED
006410 000000 SLPADDR: .WORD 0 : IN THE CURRENT TEST
006412 000000 SLPERR: .WORD 0 : CONTAINS THE LOOP ADDRESS
006414 000000 : CONTAINS THE ERROR LOOP ADDRESS

006414	000000		\$ERRPC: .WORD	0	: CONTAINS THE PC OF THE ERROR CALL
006416	000000		GOODDAT: .WORD	0	: CONTAINS THE GOOD DATA OF A TEST
006420	000000		BADDAT: .WORD	0	: CONTAINS THE BAD DATA OF A TEST
006422	000000		SWR: .WORD	0	: CONTAINS THE CURRENT VALUE OF THE FLAGS
006424	000000		SWR1: .WORD	0	
006426	000002		TPC: .WORD	END	: TEST PC FOR HARDCORE TESTS
006430	000000		RELOC: .WORD	END	: END ADDRESS OF HARDCORE
006432	000000		FILPTR: .WORD		: INDEX FOR RAD50 FILE NAME
006434	006570		OVRADR: .WORD		: START ADR FOR READ OVERLAY
006436	000000		OVRBYT: .WORD		: BYTE COUNT FOR READ OVR
006440	000000		TYPADR: .WORD		: ADDRESS OF DATA FOR TYPE CALLS
006442	000000		MODADR: .WORD		: ADR OF MODULE STRING
006444	000000		SRCADR: .WORD		: ACR FO DATA FOR LOAD WCS
006446	000000		WCSADR: .WORD		: ADR OF WCS FOR LOAD WCS
006450	000000		WCSCNT: .WORD		: WORD COUNT FOR LOAD WCS
006452	000000		STSNO: .WORD		: STS COUNT
006454	000000		IDDAT: .WORD		: DATA POINTER FOR LOAD ID
006456	000000		IDADR: .WORD		: ADDRESS OF ID REG
006460	000000		RDIDLO: .WORD		: LO 16 BITS OF READ ID DATA
006462	000000		RDIDHI: .WORD		: HI 16 BITS
006464	000000		GOTUPC: .WORD		: RECEIVED UPC FOR GETUPC
006472	002	015	\$CRLF: .BYTE	2,15,12,0	: ASCII FOR A "CRLF"
006475	000		COMSPC: MES	<, >,NB	
006502			MSG A: MES	<DATA: >,NB	
006510			MSG B: MES	<TRACE: >,NB	
006516			MSG C: MES	<?KEYBOARD ERROR: >,NB	
006534			SIXSPC: MES	< >	
006542	000000		KEYCODE: .WORD		
006544	000000		SPSW: .WORD		
006546	000001		PASCNT: .WORD	1	: USER SET PASS COUNT
006550	000000		FPYVEC: .WORD		: FLOPPY INTERRUPT VECTOR
006552	001004		LOS LNK: .WORD	F2TNO	: THIS LOCATION ONLY GETS DEFINED FOR THE MICRO DIAGNOSTIC MONITOR. IT IS USED AS A LINKAGE TO THE LOCAL TAGS OF THE MICRO DIAGNOSTIC MONITOR.
006554	000000		LOSSEC: .WORD	0	
006556	000000		SECTOR: .WORD	0	
006560	000000		FPSYNC: .WORD	0	: THIS WORD CONTAINS THE MICRO ADDRESS THAT WAS SPECIFIED IF A "SET FP" COMMAND COMMAND HAS BEEN ISSUED. IT IS USED BY THE GO CHAIN MONITOR TO SET THE SYNC POINT AT EACH NEWST STATEMENT.
006562	002746		TERMIN T: .WORD	KEYINT	: THIS LOCATION IS USED BY THE COMMAND PARSER WHEN A REPEAT FUNCTION IS SPECIFIED. IT IS ONLY DEFINED IN ESKAB.
006564	000000		MODLNK: .WORD		: THIS LOCATION IS LOADED BY THE HARDCORE MONITOR AND THE FAILCHAIN MONITOR TO POINT AT THE RAD50 LIST OF MODULE NAMES

1442

1443

1444

1445

1446

1447

1448

1449 006570 000000

1450 006572

END: .WORD 0
.BLKB 576

1451

1452

1453

1454

1455

1456

1457

1458

1459 007370 000000

1460 000001

TAGEND: .WORD
.END

;+ THE HARDCORE, GO CHAIN, AND FAIL CHAIN MONITORS ARE LOADED STARTING
; HERE. THE FIRST 512 BYTES (4 SECTORS) OF THESE FILES IS ALWAYS RESERVED
; FOR THEIR RESPECTIVE COMMON TAGS (LOCAL VARIABLES).
;-

;+ THE HARDCORE, GO CHAIN, AND FAIL CHAIN MONITORS ARE RELOADED
; STARTING HERE AND THE PARSER AND DIRECTORY PROGRAMS ARE LOADED
; STARTING HERE. IN OTHER WORDS, FROM THIS POINT TO APPROXIMATELY
; ADDRESS 23400(0) IS THE OVERLAY AREA.
;-

SYMBOL TABLE

ABELL	001134R	DCP	= 000005	HCMONI=	000000	MPC	= 000037	RADHEX=	000020
ABORT	003342R	DDP	= 000006	HCTSTR	001070R	MPF	= 001114R	RADACT=	000010
ACCMNT=	000026	DEP	= 000007	IBCLKS=	000012	MPFC	= 000026	RDIDHI	006466R
ACCST =	000027	DICMD =	001000	IBDAT =	000000	MPG	= 001112R	RDIDLO	006464R
ACCO =	000024	DIRECT=	000014	IBICT =	000013	MPGOCH=	000024	RDYIE =	000100
ACC1 =	000025	DIRERR=	000100	IBNIN =	000011	MPI	= 000036	READDS	002512R
ADAOFF=	000126	DIRSCH	001102R	IBTOD =	000001	MPS	= 000040	READMO	002270R
BADDAT	006422R	DNEIE =	C00040	ICL	= 000012	MSB	= 000022	READSC=	000004
BELFLG	001020R	DRA	= 000055	IDADR	006462R	MSBE	= 000043	RELOAD	002332R
BELL =	000020	D.SV	= 000056	IDBUS =	000051	MSF	= 001120R	RELOC	006434R
BUSOFF=	000120	END	006570R	IDCS =	173030	MSFC	= 000032	REST	001616R
BYL =	000044	ENDSPA	006374R	IDCYCL=	100000	MSG	= 001116R	RESTR	003224R
BYTCNT	001022R	ERABT =	000040	IDDAT	006460R	MSGA	= 006502R	REST2	001600R
BYU =	000045	ESP	= 000051	IDDATH=	173010	MSGB	= 006510R	REST3	001670R
B1FULL=	000004	FAD	= 000033	IDDAVL=	173006	MSGC	= 006516R	RMWRON=	000015
B1INUS=	000400	FAILCH=	000016	IDMAIN=	000200	MSGOCH=	= 000030	ROM0	= 173000
B2FULL=	000200	FCHAI1=	000020	IDP	= 000021	MSG1	= 001140R	ROM1	= 173002
B2INUS=	000040	FCHAI2=	000022	IDREGH=	000001	MSG12	= 001250R	RUNFLG=	000002
CAM =	000013	FCHR1 =	000000	IDREGL=	177777	MSG13	= 001254R	RWDQ =	000010
CCPT0 =	000200	FCHR2 =	000000	IDWRIT=	000100	MSG14	= 001264R	RXDNE	= 173014
CCPT1 =	000100	FCH1	001106R	INIT	= 010000	MSG2	= 001160R	RXVEC	= 000304
CCPT2 =	000040	FCH2	001110R	IRC	= 000020	MSG24	= 001270R	RSSET	= 000020
CCPT3 =	000020	FCT	= 000034	ISP	= 000054	MSG25	= 001302R	R6	= %000006
CDM =	000014	FILBUF	001122R	ITSTPT=	000004	MSG26	= 001334R	R7	= %000007
CEH =	000011	FILPTR	006436R	KEYBUF	001376R	MSG3	= 001200R	SAVER	003202R
CES =	000014	FILTBL	001032R	KEYCOD	006542R	MSG30	= 001344R	SAVESE	001026R
CHAR =	000002	FIRSTC=	000000	KEYERR=	020000	MSG31	= 001350R	SBC	= 000002
CHKSWI=	000023	FLPYMS=	003000	KEYINT	002746R	MSG32	= 001354R	SBH	= 000017
CH1 =	000000	FLPYON=	010000	KEYQUE=	010000	MSG4	= 001210R	SBICP	= 000036
CH2 =	000000	FLPYT1	001074R	KSP	= 000050	MSG5	= 001216R	SBIERR	= 000031
CH3 =	000000	FLPYT2	001100R	LCANWC=	000014	MSG7	= 001234R	SBIFLT	= 000033
CIA =	000056	FLPY2 =	001000	LCWRON=	000016	M256KO=	= 000116	SBIMAT	= 000035
CIB =	000000	FLPY3 =	002000	LDCONS=	000021	M4KOFF=	= 000052	SBISCM	= 000034
CLK =	000026	FLPY4 =	003000	LOADAD	001010R	M6KOFF=	= 000072	SBISIL	= 000030
CLKFST=	000010	FMH	= 000031	LOADCN=	000006	M64KOF=	= 000114	SBITO	= 000032
CLKSL0=	000020	FMIDHI=	173026	LOOP	= 000004	NER	= 000010	SBL	= 000016
CLKSTP=	000040	FMIDLO=	173024	LOSLNK	006552R	NOCHAR	= 000006	SBR	= 000046
CLRUWR=	000200	FML	= 000032	LOSS	= 000100	OFFSET	= 006370	SCBB	= 000073
CNVERT=	000007	FNM	= 000030	LOSSEC	006554R	OPENFL	= 000003	SCISPA	= 040000
COM =	100000	FPA	= 010000	LOST	= 000200	OPNFL1=	= 000011	SECTNO	006404R
COMMA	C01132R	FPDA	= 000055	MAT	= 000041	OVRADR	= 006440R	SECTOR	006556R
COMSPC	006476R	FPSYNC	006560R	MAY	= 000025	OVRBYT	= 006442R	SINST	= 000400
CONACK=	000200	FPYVEC	006550R	MAY4	= 000046	PARSE	= 001076R	SIR	= 000016
CONCM =	001000	FRO	= 000010	MAY6	= 000035	PARSER	= 000010	SIXSPC	006534R
CONID =	000003	FR1	= 000020	MAY8	= 000047	PASCNT	= 006546R	SLFTST	= 000004
COMMCR=	173032	F2SNO	001006R	MBA	= 000054	PCBB	= 000072	SLR	= 000076
COMMCS=	173034	F2TNO	001004R	MCN	= 000023	PCS	= 000003	SOMM	= 000160
CONRXD=	000005	GOCHAI=	000004	MDMTYP=	000022	PERIOD	= 001130R	SPARE1	= 173004
CONRXS=	000004	GOCHA1=	000006	MDT	= 000024	PROCEE	= 000001	SPARE2	= 173012
CONT =	004000	GOCHA2=	000012	MICFAI	001104R	PSL	= 000017	SRCADR	= 006450R
CONTXD=	000007	GOODDA	006416R	MICGO	001072R	POBR	= 000044	SSP	= 000052
CONTXS=	000006	GOSECT	001030R	MICMON	002034R	POLR	= 000074	START1	= 001452R
CPURUN=	000400	GOTUPC	006470R	MIC1FL=	002000	P1BR	= 000045	STAR12	= 001776R
CSBUS =	000050	HALTD =	000001	MIC2FL=	004000	P1LR	= 000075	STS	= 000004
CTRLC =	040000	HALTI =	000002	MNTRTN=	002000	QUEST	= 001374R	STSNO	= 006456R
DAP =	000004	HARDC =	000001	MODADR	006446R	Q.SV	= 000057	SUBTST	= 006400R
DBP =	000010	HARDCO	001066R	MODLNK	006564R	RADGET	= 000010	SWR	= 006426R

ZZ-ESKAB-14.0 VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 2 29-OCT-82
VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200 29-OCT-82 10:28 PAGE 27-2
SYMBOL TABLE

E 8

Fiche 1 Frame E8

Sequence 95

SWR1	006430R	TOIDHI= 173022	USC = 000001	\$CPUTR 002402R	\$RD IDR 004360R
TAGEND	007370R	TOIDLO= 173020	USCADR= 000042	\$CRLF 006472R	\$READO 004456R
TBDAT =	000020	TPC 006432R	USCBRK= 000041	\$DONE 003370R	\$RNGBE 004554R
TBER0 =	000022	TPCINI= 000034	USCDAT= 000043	\$DONEM 003376R	\$SBCCCL 004624R
TBER1 =	000023	TRAPVE= 000034	USCSTK= 000040	SENCTR 004650R	\$SCTNO 006402R
TBM =	000015	TREAD = 000002	USP = 000053	SENDAD= 020400	\$STSCL 004672R
TEMP =	006534R	TRS = 000027	VBCLK = 000001	SERFLG 006406R	\$TBSY = 000005
TEMPO =	000060	TSTMFG= 000024	VBCTRL= 173036	SERRPC 006414R	\$TCTC = 000006
TEMP1 =	000061	TSTSPA= 020000	VBLOAD= 000002	\$FAILC 003722R	\$TEMP1= 006542R
TEMP2 =	000062	TWOSPC 001136R	VBUS = 000052	\$FER = 000001	\$TER = 000007
TEMP3 =	000063	TWRITE= 000001	VECT = 000015	\$FNF = 000002	\$TMP0 001012R
TEMP4 =	000064	TXRDY = 173015	W = 006370R	\$FNRF = 000003	\$TMP1 001014R
TEMP5 =	000065	TXVEC = 000300	WCS = 000002	\$FOR = 000004	\$TMP2 001016R
TEMP6 =	000066	TYPADR 006444R	WCSCNT 006452R	\$GETUP 003630R	\$TRAP 003246R
TEMP7 =	000067	TYPBUF 001400R	WCS2K = 000042	\$LDWCS 004044R	\$TRPAD 003266R
TEMP8 =	000070	TYPESD 005310R	WRITSC= 000005	\$LOADI 004132R	\$STSTNM 006372R
TEMP9 =	000071	TYPLNG 00024R	WW = 000000R	\$LPADR 006410R	\$TYPE 004724R
TERMIN	006562R	TYPPAS 003102R	X = 000100	\$LPERR 006412R	\$TYPEB 005262R
TESTNO	006376R	TYPSEC 006270R	XX = 000176	\$MICMO 004200R	\$TYPED 005302R
TESTST=	000002	TYP1 = 000012	Y = 006566R	\$OPNFI 004214R	\$TYPER 005372R
TINIT =	000000	TYP2 = 000013	Z = 000002	SPASS 00637CR	\$TYPES 005272R
TITLE	001360R	UBA = 000053		SPSW 006544R	\$TYPMO 005666R
TMERTR=	000017	UPC12 = 001000			

. ABS. 000000 000
007372 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 23492 WORDS (92 PAGES)

DYNAMIC MEMORY: 20060 WORDS (77 PAGES)

ELAPSED TIME: 00:03:18

ESKAB,ESKAB/-SP=ESKABMAC.MLB/ML,ESKAB.MAC

ZZ-ESKAB-14.0 Directory

F 8
29-OCT-82

Fiche 1 Frame F8

Sequence 96

Fiche	Frame	Sequence			
1	B1	1	Documentation		
1	J4	48	TABLE OF CONTENTS		29-OCT-82
1	K4	49	VAX 11/780 MICRO DIAGNOSTIC MON MACRO M1200	2	29-OCT-82

B 1 Documentation
C 1 Documentation
D 1 Documentation
E 1 Documentation
F 1 Documentation
G 1 Documentation
H 1 Documentation
I 1 Documentation
J 1 Documentation
K 1 Documentation
L 1 Documentation
M 1 Documentation
N 1 Documentation
B 2 Documentation
C 2 Documentation
D 2 Documentation
E 2 Documentation
F 2 Documentation
G 2 Documentation
H 2 Documentation
I 2 Documentation
J 2 Documentation
K 2 Documentation
L 2 Documentation
M 2 Documentation
N 2 Documentation
B 3 Documentation
C 3 Documentation
D 3 Documentation
E 3 Documentation
F 3 Documentation
G 3 Documentation
H 3 Documentation
I 3 Documentation
J 3 Documentation
K 3 Documentation
L 3 Documentation
M 3 Documentation
N 3 Documentation
B 4 Documentation
C 4 Documentation
D 4 Documentation
E 4 Documentation
F 4 Documentation
G 4 Documentation
H 4 Documentation
I 4 Documentation
J 4 Documentation
K 4 TABLE OF CONTENTS
L 4 VAX 11/780 MICRO DIAGNOSTIC MO
M 4 VAX 11/780 MICRO DIAGNOSTIC MO
N 4 VAX 11/780 MICRO DIAGNOSTIC MO
B 5 VAX 11/780 MICRO DIAGNOSTIC MO
C 5 VAX 11/780 MICRO DIAGNOSTIC MO
D 5 VAX 11/780 MICRO DIAGNOSTIC MO
E 5 VAX 11/780 MICRO DIAGNOSTIC MO
F 5 VAX 11/780 MICRO DIAGNOSTIC MO
G 5 VAX 11/780 MICRO DIAGNOSTIC MO
H 5 VAX 11/780 MICRO DIAGNOSTIC MO
I 5 VAX 11/780 MICRO DIAGNOSTIC MO

J 5 VAX 11/780 MICRO DIAGNOSTIC MO
K 5 VAX 11/780 MICRO DIAGNOSTIC MO
L 5 VAX 11/780 MICRO DIAGNOSTIC MO
M 5 VAX 11/780 MICRO DIAGNOSTIC MO
N 5 VAX 11/780 MICRO DIAGNOSTIC MO
B 6 VAX 11/780 MICRO DIAGNOSTIC MO
C 6 VAX 11/780 MICRO DIAGNOSTIC MO
D 6 VAX 11/780 MICRO DIAGNOSTIC MO
E 6 VAX 11/780 MICRO DIAGNOSTIC MO
F 6 VAX 11/780 MICRO DIAGNOSTIC MO
G 6 VAX 11/780 MICRO DIAGNOSTIC MO
H 6 VAX 11/780 MICRO DIAGNOSTIC MO
I 6 VAX 11/780 MICRO DIAGNOSTIC MO
J 6 VAX 11/780 MICRO DIAGNOSTIC MO
K 6 VAX 11/780 MICRO DIAGNOSTIC MO
L 6 VAX 11/780 MICRO DIAGNOSTIC MO
M 6 VAX 11/780 MICRO DIAGNOSTIC MO
N 6 VAX 11/780 MICRO DIAGNOSTIC MO
B 7 VAX 11/780 MICRO DIAGNOSTIC MO
C 7 VAX 11/780 MICRO DIAGNOSTIC MO
D 7 VAX 11/780 MICRO DIAGNOSTIC MO
E 7 VAX 11/780 MICRO DIAGNOSTIC MO
F 7 VAX 11/780 MICRO DIAGNOSTIC MO
G 7 VAX 11/780 MICRO DIAGNOSTIC MO
H 7 VAX 11/780 MICRO DIAGNOSTIC MO
I 7 VAX 11/780 MICRO DIAGNOSTIC MO
J 7 VAX 11/780 MICRO DIAGNOSTIC MO
K 7 VAX 11/780 MICRO DIAGNOSTIC MO
L 7 VAX 11/780 MICRO DIAGNOSTIC MO
M 7 VAX 11/780 MICRO DIAGNOSTIC MO
N 7 VAX 11/780 MICRO DIAGNOSTIC MO
B 8 VAX 11/780 MICRO DIAGNOSTIC MO
C 8 VAX 11/780 MICRO DIAGNOSTIC MO
D 8 VAX 11/780 MICRO DIAGNOSTIC MO
E 8 VAX 11/780 MICRO DIAGNOSTIC MO
F 8 VAX 11/780 MICRO DIAGNOSTIC MO

8 Directory